9



ALAGAPPA UNIVERSITY

(Reaccredited with 'A' Grade by NAAC)

KARAIKUDI-630 003, TAMILNADU



DIRECTORATE OF DISTANCE EDUCATION

(Recognized by Distance Education Council (DEC), New Delhi)

MBA (System Management)



Paper - 4.4

DATAMINING AND WAREHOUSING

ALAGAPPA UNIVERSITY

(Accredited with 'A' Grade by NAAC)

KARAIKUDI - 630 003. TAMILNADU.

DIRECTORATE OF DISTANCE EDUCATION

(Recognized by Distance Education Council (DEC), New Delhi)

MBA (System Management)



PAPER - 4.4

DATAMINING AND WAREHOUSING

PAPER 4.4: DATA WAREHOUSING AND MINING

UNIT I

Introduction : Data mining – Kinds of data for mining, Data mining Functionalities, Classification of Data mining systems, Major issues on Data mining, Introduction to OLAP, OLAP technology for Data mining, Data warehousing, Data warehousing to Data Mining, Optimizing data for mining, Data preprocessing

UNIT II

Data Mining Primitives : Data mining Query language, Association Rules in large Data mining, KDD Process, Fuzzy sets and logic, Classification and Prediction : Information retrieval, Dimensional Modeling of Data, Pattern Matching, Estimation Error – Em, MLE.

UNIT III

Models based on Summarization : Bayes Theorem, Chi squared Statistics Regression, Decision Tree, Neural Networks, Genetic Algorithms, Cluster Analysis – Outlier, Cluster Vs Classification, Clustering issues, impact of Outliers on clustering, clustering problems, Clustering Approaches.

UNIT IV

Clustering Algorithms : Hierarchical algorithm – Single Link, MST Single Link, Complete Link, Average Link, Dendrogram. Partitional Algorithm – MST, Squared Error, K-Means, Nearest Neighbor, PAM, BEA, GA, Categorical algorithm, Large Database.

UNIT V

Web mining : Introduction, Web data, Web Knowledge Mining Taxonomy, Web Content mining, Web Usage mining Research, Ontology based web mining Research, Web mining Applications.

TEXT BOOKS:

1. Jaiwei Han and Micheline Kamber, "Data mining concepts and techniques", Harcourt India Pvt Ltd, 2001.
2. Sam Anahory and Dennis Murray, "Data Warehousing in the Real World", Pearson Education 2005.

REFERENCE BOOKS:

1. Margaret H Dunham, "Data Mining Introductory and Advanced topics", Pearson Education, 2003.
2. Paulraj Ponnaiah, "Data Warehousing Fundamentals", Wiley Student Edition, 2007.
3. Ralph Kimball, "The Data Warehouse Life Cycle Tool Kit", Wiley Student 2nd Edition, 2008.

Course Material Prepared By:-

Mrs.A. Padmapriya,
Assistant Professor ,
Department of Computer Science and Engineering,
Alagappa University,
Karaikudi – 630 003.

UNIT I AN INTRODUCTION TO DATA MINING

Structure

I.0 Objectives

In this unit we are going to discuss about the basic concepts related to data mining. This unit also elaborates how data mining acts as the core process of knowledge discovery from databases. It also discusses about OLAP and issues related to Data mining.

At the end of this unit we can

- Understand the concept of data mining
- Identify the kinds of information collected for Data mining
- Gain knowledge about the Knowledge Discovery in Databases
- Optimize the data used for mining

I.1 : Introduction to Data Mining

I.1.1 Introduction

We are in an age often referred to as the information age. In this information age, because we believe that information leads to power and

success, and thanks to sophisticated technologies such as computers, satellites, etc., we have been collecting tremendous amounts of information. Initially, with the advent of computers and means for mass digital storage, we started collecting and storing all sorts of data, counting on the power of computers to help sort through this amalgam of information.

Unfortunately, these massive collections of data stored on disparate structures very rapidly became overwhelming. This initial chaos has led to the creation of structured databases and database management systems (DBMS). The efficient database management systems have been very important assets for management of a large corpus of data and especially for effective and efficient retrieval of particular information from a large collection whenever needed. The rise / increased used of database management systems has also contributed to recent massive gathering of all sorts of information.

Today, we have far more information than we can handle: from business transactions and scientific data, to satellite pictures, text reports and military intelligence. Information retrieval is simply not enough anymore for decision-making. Confronted with huge collections of data, we have now created new needs to help us make better managerial choices. These needs are automatic summarization of data, extraction of the "essence" of information stored, and the discovery of patterns in raw data.

I.1.2 Importance of Data Mining

The major reason that data mining has attracted a great deal of attention in the information industry in recent years is due to the wide availability of huge amounts of data and the imminent (about to happen) need for turning such data into useful information and knowledge. The information and knowledge gained can be used for applications ranging from business management, production control, and market analysis, to engineering design and science exploration. Data mining can be viewed as a result of the natural evolution of information technology.

→ ***Data collection and database creation***

- 1960s and earlier.

- Database and information technology has been evolving systematically from primitive file processing systems to sophisticated and powerful database systems.

→ ***Database management systems***

- 1970s – Early 1980s.
- Database system has progressed from early hierarchical and network database systems to the relational database systems, data modeling tools and indexing and data organization techniques.
- Users gained convenient and flexible data access through query languages, user interfaces, optimized query processing and transaction management.
- OLTP (On-Line Transaction Processing) – where query is viewed as a read-only transaction – is a major tool for efficient storage, retrieval and management of large amounts of data.

→ ***Advanced database systems***

- Mid 1980s – present
- Database technology has been characterized by the popular adoption of relational technology.
- These new and powerful systems employ advanced data models such as extended-relational object-oriented, object-relational and deductive models.
- Application oriented database systems including spatial, temporal, multimedia, active and scientific databases, knowledge bases and office information bases, have flourished.

→ ***Web-based database systems***

- Mid 1990s – present
- Heterogeneous database systems and Internet-based global information systems such as World Wide Web (WWW) have also emerged and play a vital role in the information industry.
- XML based database systems and web mining concepts are introduced

→ ***Data warehousing and Data Mining***

- Late 1980s – present
- The steady and amazing progress of computer hardware technology in the past 3 decades has led to large supplies of powerful and affordable computers, data collection equipment and storage media.
- This makes a huge number of databases and information repositories available for transaction management, information retrieval and data analysis.

Data can now be stored in many different types of data bases.

"***Data warehouse***" is a repository of multiple heterogeneous data sources, organized under a unified schema at a single site in order to facilitate management decision making. Data warehouse technology includes data cleansing, data integration and On-Line Analytical Processing (OLAP) i.e. analysis techniques with functionalities such as summarization, consolidation and aggregation, as well as the ability to view information from different angles. Although OLAP tools support multidimensional analysis and decision making, additional data analysis tools are required for in-depth analysis.

The abundance of data, coupled with the need for powerful data analysis tools has been described as a ***data rich but information poor*** situation. As a result, data collected in large databases and become "***data tombs***" - data archives that are seldom visited. Consequently, important decisions are often made based not on the information-rich data stored in databases but rather on the decision maker's intuition, simply because the decision maker does not have the tools to extract the valuable knowledge embedded in the vast amounts of data.

Data mining tools perform data analysis and may uncover important data patterns, contributing greatly to business greatly to business strategies, knowledge bases and scientific and medical research. The widening gap between data and information calls for a systematic development of data mining tools that will turn data tombs into "***golden nuggets***" of knowledge.

I.1.3 What is Data mining?

- Extracting or "mining" knowledge from large amounts of data.
- Data – driven discovery and modeling of hidden patterns in large volumes of data.
- Extraction of implicit, previously unknown and unexpected, potentially, extremely useful information from data.

There are many other terms carrying a similar or slightly different meaning to data mining, such as knowledge mining from databases, knowledge extraction, data/pattern analysis, data archeology and data dredging.

I.1.4 Data mining definitions

Some of the definitions of Data Mining, or Knowledge Discovery in Databases are:

"***Extraction of interesting information or patterns from data in large databases is known as data mining.***"

Data Mining refers to "***using a variety of techniques to identify nuggets of information or decision-making knowledge in bodies of data and extracting these in such a way that they can be put to use in the areas such as decision support, prediction, forecasting and estimation. The data is often voluminous, but as it stands of low value as no direct use can be made of it; it is the hidden information in the data that is useful***"

According to William J. Frawley, Gregory Piatetsky-Shapiro and Christopher J. Matheus, "***Data Mining, or Knowledge Discovery in Databases (KDD) as it is also known, is the non-trivial extraction of implicit, previously unknown and potentially useful information from data. This encompasses a number of different technical approaches, such as clustering, data summarization, learning classification rules, finding dependency networks, analyzing changes, and detecting anomalies***".

According to Marcel Holshemier and Arno Siebes "***Data Mining is the search for relationships and global patterns that exist in large databases but are hidden among the vast amount of data such as a relationship between patient data and their medical diagnosis. These relationships represent valuable knowledge about the database and the objects in the database and, if the database is a faithful mirror, of the real world registered by the database***".

I.2 Kinds of Information

We have been collecting a myriad of data, from simple numerical measurements and text documents, to more complex information such as spatial data, multimedia channels, and hypertext documents. Here is a non-exclusive list of a variety of information collected in digital form in databases and in flat files.

- **Business transactions**:
  - Every transaction in the business industry is (often) "memorized" for perpetuity
  - Such transactions are usually time related and can be *inter-business deals* such as purchases, exchanges, banking, stock, etc., or *intra-business operations* such as management of in-house wares and assets.
  - Large departmental stores, store millions of transactions daily representing often terabytes of data.
  - Storage space is not the major problem, as the price of hard disks is continuously dropping, but the effective use of the data in a reasonable time frame for competitive decision-making is definitely the most important problem to solve for businesses that struggle to survive in a highly competitive world.

- **Scientific data**:
  - Irrespective of the place, (whether in a Swiss nuclear accelerator laboratory counting particles, in the Canadian forest studying readings from a grizzly bear radio collar, on a South Pole iceberg gathering data about oceanic activity, or in an American university investigating human psychology) our society is amassing colossal amounts of scientific data that need to be analyzed.

- Unfortunately, we can capture and store more new data faster than we can analyze the old data already accumulated.

- **Medical and personal data**:
  - From government census to personnel and customer files, very large collections of information are continuously gathered about individuals and groups.
  - Governments, companies and organizations such as hospitals, are stockpiling very important quantities of personal data to help them manage human resources, better understand a market, or simply assist clientele.
  - Regardless of the privacy issues this type of data often reveals, this information is collected, used and even shared.
  - When correlated with other data this information can shed light on customer behavior and the like.

- **Surveillance video and pictures**:
  - With the amazing collapse of video camera prices, video cameras are becoming ubiquitous.
  - Video tapes from surveillance cameras are usually recycled and thus the content is lost.
  - However, there is a tendency today to store the tapes and even digitize them for future use and analysis.

- **Satellite sensing**:
  - There are a countless number of satellites around the globe: some are geo-stationary above a region, and some are orbiting around the Earth, but all are sending a non-stop stream of data to the surface.
  - NASA, which controls a large number of satellites, receives more data every second than what all NASA researchers and engineers can cope with.
  - Many satellite pictures and data are made public as soon as they are received in the hopes that other researchers can analyze them.

- **Games**:
    - ➢ Our society is collecting a tremendous amount of data and statistics about games, players and athletes.
    - ➢ From hockey scores, basketball passes and car-racing lapses, to swimming times, boxer's pushes and chess positions, all the data are stored.
    - ➢ Commentators and journalists are using this information for reporting, but trainers and athletes would want to exploit this data to improve performance and better understand opponents.

- **Digital media**:
    - ➢ The proliferation of cheap scanners, desktop video cameras and digital cameras is one of the causes of the explosion in digital media repositories.
    - ➢ In addition, many radio stations, television channels and film studios are digitizing their audio and video collections to improve the management of their multimedia assets.
    - ➢ Associations such as the NHL (National Hockey League) and the NBA (National Basketball League) have already started converting their huge game collection into digital forms.

- **CAD and Software engineering data**:
    - ➢ There are a multitude of Computer Assisted Design (CAD) systems for architects to design buildings or engineers to conceive system components or circuits.
    - ➢ These systems are generating a tremendous amount of data. Moreover, software engineering is a source of considerable similar data with code, function libraries, objects, etc., which need powerful tools for management and maintenance.

- **Virtual Worlds**:
    - ➢ There are many applications making use of three-dimensional virtual spaces.
    - ➢ These spaces and the objects they contain are described with special languages such as VRML.

- Ideally, these virtual spaces are described in such a way that they can share objects and places.
- There is a remarkable amount of virtual reality object and space repositories available.
- Management of these repositories as well as content-based search and retrieval from these repositories are still research issues, while the size of the collections continues to grow.

- **Text reports and memos (e-mail messages)**:
  - Most of the communications within and between companies or research organizations or even private people, are based on reports and memos in textual forms often exchanged by e-mail.
  - These messages are regularly stored in digital form for future use and reference creating formidable digital libraries.

- **The World Wide Web repositories**:
  - Since the inception of the World Wide Web in 1993, documents of all sorts of formats, content and description have been collected and inter-connected with hyperlinks making it the largest repository of data ever built.
  - Despite its dynamic and unstructured nature, its heterogeneous characteristic, and its very often redundancy and inconsistency, the World Wide Web is the most important data collection regularly used for reference because of the broad variety of topics covered and the infinite contributions of resources and publishers.
  - Many believe that the World Wide Web will become the compilation of human knowledge.

I.3 Data Mining

With the enormous amount of data stored in files, databases, and other repositories, it is increasingly important, if not necessary, to develop powerful means for analysis and perhaps interpretation of such data and for the extraction of interesting knowledge that could help in decision-making.

I.3.1 The Knowledge Discovery in Databases process

Data Mining, also popularly known as Knowledge Discovery in Databases (KDD), refers to the nontrivial extraction of implicit, previously unknown and potentially useful information from data in databases. While data mining and knowledge discovery in databases (or KDD) are frequently treated as synonyms, data mining is actually part of the knowledge discovery process. The following figure (Figure I.3.1) shows data mining as a step in an iterative knowledge discovery process.

The Knowledge Discovery in Databases process comprises of a few steps leading from raw data collections to some form of new knowledge. The iterative process consists of the following steps:

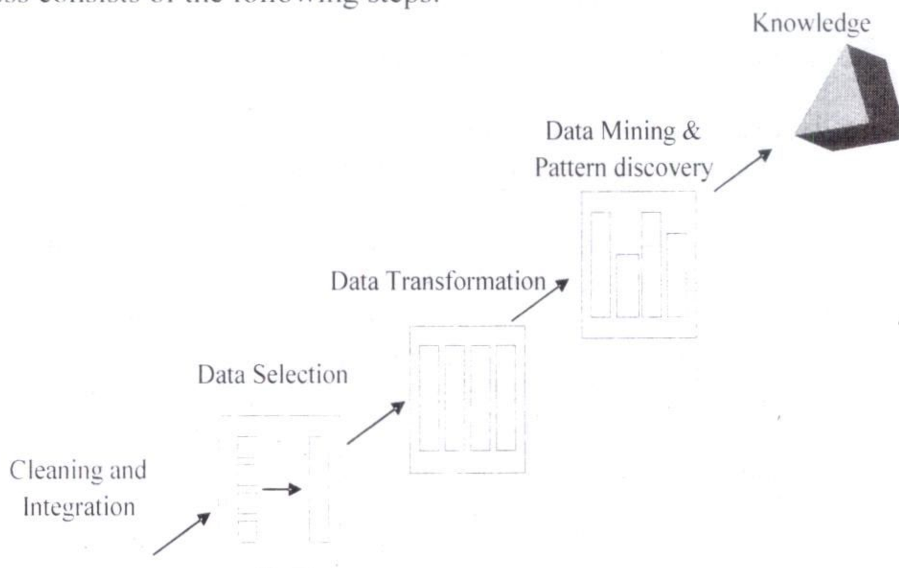


Figure. I.3.1 : An overview of the steps that compose the KDD process

- ***Data cleaning***: also known as data cleansing, it is a phase in which noise data and irrelevant data are removed from the collection.
- ***Data integration***: at this stage, multiple data sources, often heterogeneous, may be combined in a common source.

- ***Data selection***: at this step, the data relevant to the analysis is decided on and retrieved from the data collection.
- ***Data transformation***: also known as data consolidation, it is a phase in which the selected data is transformed into forms appropriate for the mining procedure.
- ***Data mining***: it is the crucial step in which clever techniques are applied to extract patterns potentially useful.
- ***Pattern evaluation***: in this step, strictly interesting patterns representing knowledge are identified based on given measures.
- ***Knowledge representation***: is the final phase in which the discovered knowledge is visually represented to the user. This essential step uses visualization techniques to help users understand and interpret the data mining results.

It is common to combine some of these steps together. For instance, data cleaning and data integration can be performed together as a pre-processing phase to generate a data warehouse. Data selection and data transformation can also be combined where the consolidation of the data is the result of the selection, or, as for the case of data warehouses, the selection is done on transformed data. The KDD is an iterative process. Once the discovered knowledge is presented to the user, the evaluation measures can be enhanced, the mining can be further refined, new data can be selected or further transformed, or new data sources can be integrated, in order to get different, more appropriate results.

Data mining derives its name from the similarities between searching for valuable information in a large database and mining rocks for a vein of valuable ore. Both imply either sifting through a large amount of material or ingeniously probing the material to exactly pinpoint where the values reside. It is, however, a misnomer, since mining for gold in rocks is usually called "gold mining" and not "rock mining", thus by analogy, data mining should have been called "knowledge mining" instead.

Nevertheless, data mining became the accepted customary term, and very rapidly a trend that even overshadowed more general terms such as knowledge discovery in databases (KDD) that describe a more complete process. Other

similar terms referring to data mining are: data dredging, knowledge extraction and pattern discovery. -

I.3.2 Architeçture of a typical Data Mining system

Data Mining is the process of discovering interesting knowledge form large amounts of data stored either in databases, data warehouses or other information repositories. Based on this view, the architecture of a typical data mining system may have the following major components.

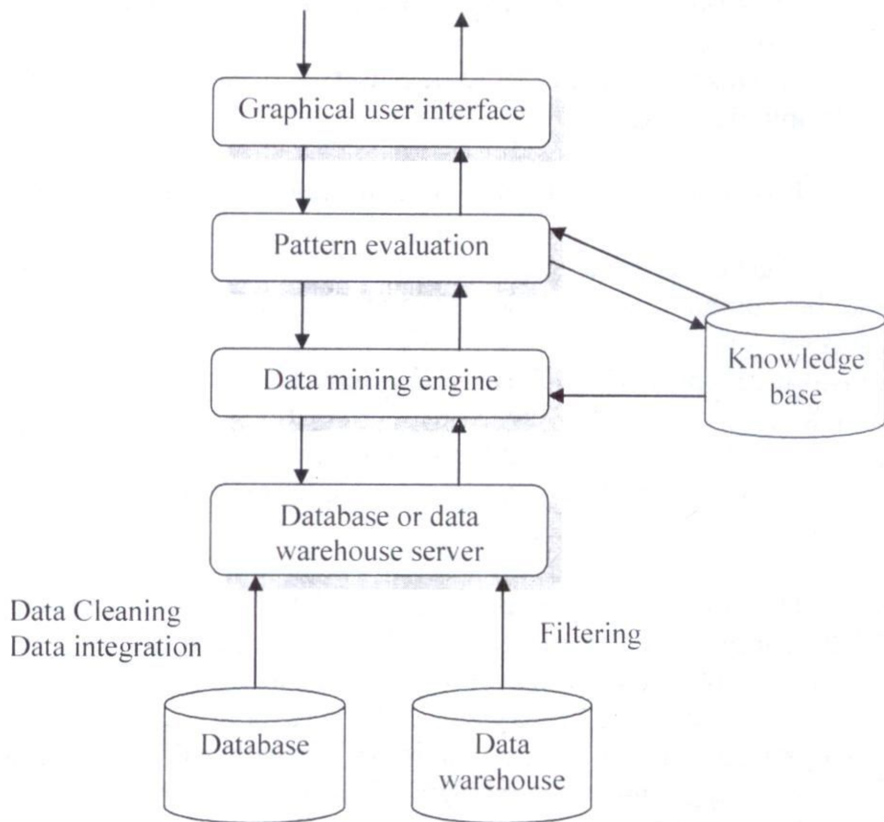


Figure I.3.2. Architecture of a typical data mining system

- ***Database, data warehouse or other information repository***
  This is one or a set of databases, data warehouses, spreadsheets or other kinds of information repositories. Data cleaning and data integration techniques may be performed on the data.

- ***Database or data warehouse server***
  The database or data warehouse server is responsible for fetching the relevant data, based on the user's data mining request.

- ***Knowledge base***
  This is the domain knowledge that is used to guide the search or evaluate the interestingness of resulting patterns. Such knowledge can include concept hierarchies (used to organize attributes into different levels of abstraction), user beliefs (used to assess the pattern's interestingness based on its unexpectedness), constraints or thresholds and metadata (e.g., describing data from multiple heterogeneous sources).

- ***Data mining engine***
  This is essential to the data mining system and ideally consists of a set of functional modules for tasks such as characterization, association, classification, cluster analysis, and evolution and deviation analysis.

- ***Pattern evaluation module***
  This component typically employs interestingness measure and interacts with the data mining modules so as to focus the search towards interesting patterns. For efficient data mining, it is highly recommended to push the evaluation of pattern interestingness as deep as possible into the mining process so as to confine the search to only the interesting patterns.

- ***Graphical user interface***
  This module communicates between the users and the data mining system, allowing the user to interact with the system by specifying a data mining query or task, providing information to help focus the search, and performing exploratory data mining based on the intermediate data mining results. In addition, this component allows the user to browse

database and data warehouse schemas or data structures, evaluate mined patterns and visualize the patterns in different forms.

From a data warehouse perspective, data mining can be viewed as an advanced stage of on-line analytical processing (OLAP). However, data mining goes far beyond the narrow scope of summarization-style analytical processing of data warehouse systems by incorporating more advanced techniques for data understanding.

1.4 Kinds of Data for mining

In principle, data mining is not specific to one type of media or data. Data mining should be applicable to any kind of information repository. However, algorithms and approaches may differ when applied to different types of data. Indeed, the challenges presented by different types of data vary significantly.

Data mining is being put into use and studied for

- Flat files
- Relational databases
- Data Warehouses
- Transaction Databases
- Multimedia Databases
- Spatial Databases
- Time-Series Databases
- World Wide Web

Here are some examples in more detail:

Flat files

→ Flat files are actually the most common data source for data mining algorithms, especially at the research level.

→ Flat files are simple data files in text or binary format with a structure known by the data mining algorithm to be applied.

→ The data in these files can be transactions, time-series data, scientific measurements, etc.

Relational Databases:

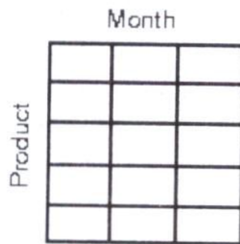
→ Briefly, a relational database consists of a set of tables containing either values of entity attributes, or values of attributes from entity relationships.

- → Tables have columns and rows, where columns represent attributes and rows represent tuples.
- → A tuple in a relational table corresponds to either an object or a relationship between objects and is identified by a set of attribute values representing a unique key.
- → The most commonly used query language for relational database is SQL, which allows retrieval and manipulation of the data stored in the tables, as well as the calculation of aggregate functions such as average, sum, min, max and count.
- → Data mining algorithms using relational databases can be more versatile than data mining algorithms specifically written for flat files, since they can take advantage of the structure inherent to relational databases.
- → While data mining can benefit from SQL for data selection, transformation and consolidation, it goes beyond what SQL could provide, such as predicting, comparing, detecting deviations, etc.

Data Warehouses

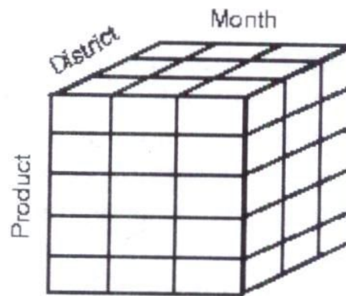
- → A data warehouse as a storehouse is a repository of data collected from multiple data sources (often heterogeneous) and is intended to be used as a whole under the same unified schema.
- → A data warehouse gives the option to analyze data from different sources under the same roof.
- → Data from the different stores would be loaded, cleaned, transformed and integrated together.
- → To facilitate decision-making and multi-dimensional views, data warehouses are usually modeled by a multi-dimensional data structure.
- → Data cubes are used to present the data in multi-dimensional views. A cube contains cells that store values of some aggregate measures and special cells that store summations along dimensions. Each dimension of the data cube contains a hierarchy of values for one attribute.

Two-Dimensional Spreadsheet

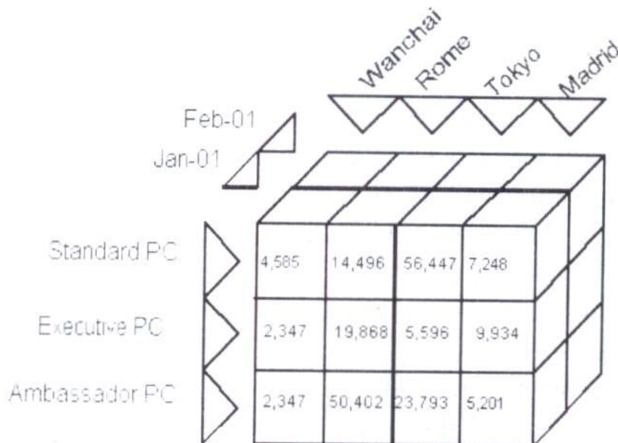


Sales data for each district is in a separate spreadsheet

Multidimensional Array



Sales data for all districts is in a single array



→ Because of their structure, the pre-computed summarized data they contain and the hierarchical attribute values of their dimensions, data cubes are well suited for fast interactive querying and analysis of data at different conceptual levels, known as On-Line Analytical Processing (OLAP).

Transaction Databases

→ A transaction database is a set of records representing transactions, each with a time stamp, an identifier and a set of items.

- → Associated with the transaction files could also be descriptive data for the items.
- → Since relational databases do not allow nested tables (i.e. a set as attribute value), transactions are usually stored in flat files or stored in two normalized transaction tables, one for the transactions and one for the transaction items.
- → One typical data mining analysis on such data is the so-called market basket analysis or association rules in which associations between items occurring together or in sequence are studied.

Multimedia Databases

- → Multimedia databases include video, images, and audio and text media.
- → They can be stored on extended object-relational or object-oriented databases, or simply on a file system.
- → Multimedia is characterized by its high dimensionality, which makes data mining even more challenging.
- → Data mining from multimedia repositories may require computer vision, computer graphics, image interpretation, and natural language processing methodologies.

Spatial Databases

- → Spatial databases are databases that, in addition to usual data, store geographical information like maps, and global or regional positioning.
- → Such spatial databases present new challenges to data mining algorithms

Time-Series Databases

- → Time-series databases contain time related data such stock market data or logged activities.
- → These databases usually have a continuous flow of new data coming in, which sometimes causes the need for a challenging real time analysis.
- → Data mining in such databases commonly includes the study of trends and correlations between evolutions of different variables, as well as the prediction of trends and movements of the variables in time.

World Wide Web

- → The World Wide Web is the most heterogeneous and dynamic repository available.

→ A very large number of authors and publishers are continuously contributing to its growth and metamorphosis, and a massive number of users are accessing its resources daily.

→ Data in the World Wide Web is organized in inter-connected documents. These documents can be text, audio, video, raw data, and even applications.

→ Conceptually, the World Wide Web is comprised of three major components:

   - *The content of the Web* : which encompasses documents available;
   - *The structure of the Web* : which covers the hyperlinks and the relationships between documents; and
   - *The usage of the web* : describing how and when the resources are accessed.

→ A fourth dimension can be added relating the dynamic nature or evolution of the documents.

→ Data mining in the World Wide Web, or web mining, tries to address all these issues and is often divided into web content mining, web structure mining and web usage mining.

1.5 Data mining Functionalities and Classification

The kinds of patterns that can be discovered depend upon the data mining tasks employed. By and large, there are two types of data mining tasks: ***descriptive data mining tasks*** that describe the general properties of the existing data, and ***predictive data mining tasks*** that attempt to do predictions based on inference on available data.

1.5.1 Data Mining Functionalities

The data mining functionalities and the variety of knowledge they discover are briefly presented in the following list:

Characterization

Data characterization is a summarization of general features of objects in a target class, and produces what is called characteristic rules. The data relevant to a user-specified class are normally retrieved by a database query and run through a summarization module to extract the essence of the data at different

levels of abstractions, normally using data cubes. Note that with a data cube containing summarization of data, simple OLAP operations fit the purpose of data characterization.

Discrimination:

Data discrimination produces what are called discriminated rules and is basically the comparison of the general features of objects between two classes referred to as the target class and the contrasting class. For example, one may want to compare the general characteristics of the customers who rented more than 30 movies in the last year with those whose rental account is lower than 5. The techniques used for data discrimination are very similar to the techniques used for data characterization with the exception that data discrimination results include comparative measures.

Association analysis:

Association analysis is the discovery of what are commonly called association rules. It studies the frequency of items occurring together in transactional databases, and based on a threshold called support, identifies the frequent item sets. Another threshold, confidence, which is the conditional probability than an item appears in a transaction when another item appears, is used to pinpoint association rules. Association analysis is commonly used for market basket analysis.

Classification:

Classification analysis is the organization of data in given classes. Also known as supervised classification, the classification uses given class labels to order the objects in the data collection. Classification approaches normally use a training set where all objects are already associated with known class labels. The classification algorithm learns from the training set and builds a model. The model is used to classify new objects. The classification analysis would generate a model that could be used to either accept or reject credit requests in the future.

Prediction:

Prediction has attracted considerable attention given the potential implications of successful forecasting in a business context. There are two major types of predictions: one can either try to predict some unavailable data values or

pending trends, or predict a class label for some data. The latter is tied to classification. Once a classification model is built based on a training set, the class label of an object can be foreseen based on the attribute values of the object and the attribute values of the classes. Prediction is however more often referred to the forecast of missing numerical values, or increase/ decrease trends in time related data. The major idea is to use a large number of past values to consider probable future values.

Clustering:

Similar to classification, clustering is the organization of data in classes. However, unlike classification, in clustering, class labels are unknown and it is up to the clustering algorithm to discover acceptable classes. Clustering is also called unsupervised classification, because the classification is not dictated by given class labels. There are many clustering approaches all based on the principle of maximizing the similarity between objects in a same class (intra-class similarity) and minimizing the similarity between objects of different classes (inter-class similarity).

- ***Outlier analysis***: Outliers are data elements that cannot be grouped in a given class or cluster. Also known as exceptions or surprises, they are often very important to identify. While outliers can be considered noise and discarded in some applications, they can reveal important knowledge in other domains, and thus can be very significant and their analysis valuable.
- ***Evolution and deviation analysis***: Evolution and deviation analysis pertain to the study of time related data that changes in time. Evolution analysis models evolutionary trends in data, which consent to characterizing, comparing, classifying or clustering of time related data. Deviation analysis, on the other hand, considers differences between measured values and expected values, and attempts to find the cause of the deviations from the anticipated values.

It is common that users do not have a clear idea of the kind of patterns they can discover or need to discover from the data at hand. It is therefore important to have a versatile and inclusive data mining system that allows the

discovery of different kinds of knowledge and at different levels of abstraction. This also makes interactivity an important attribute of a data mining system.

Data mining allows the discovery of knowledge potentially useful and unknown. Whether the knowledge discovered is new, useful or interesting, is very subjective and depends upon the application and the user. It is certain that data mining can generate, or discover, a very large number of patterns or rules. In some cases the number of rules can reach the millions. One can even think of a meta-mining phase to mine the oversized data mining results. To reduce the number of patterns or rules discovered that have a high probability to be non-interesting, one has to put a measurement on the patterns. However, this raises the problem of completeness. The user would want to discover all rules or patterns, but only those that are interesting. The measurement of how interesting a discovery is, often called interestingness, can be based on quantifiable objective elements such as validity of the patterns when tested on new data with some degree of certainty, or on some subjective depictions such as understandability of the patterns, novelty of the patterns, or usefulness.

Discovered patterns can also be found interesting if they confirm or validate a hypothesis sought to be confirmed or unexpectedly contradict a common belief. This brings the issue of describing what is interesting to discover, such as meta-rule guided discovery that describes forms of rules before the discovery process, and interestingness refinement languages that interactively query the results for interesting patterns after the discovery phase. Typically, measurements for interestingness are based on thresholds set by the user. These thresholds define the completeness of the patterns discovered.

Identifying and measuring the interestingness of patterns and rules discovered, or to be discovered is essential for the evaluation of the mined knowledge and the KDD process as a whole. While some concrete measurements exist, assessing the interestingness of discovered knowledge is still an important research issue.

I.5.2 Data mining classifications

There are many data mining systems available or being developed. Some are specialized systems dedicated to a given data source or are confined to

limited data mining functionalities, other are more versatile and comprehensive. Data mining systems can be categorized according to various criteria among other classification are the following:

- ***Classification according to the type of data source mined***: this classification categorizes data mining systems according to the type of data handled such as spatial data, multimedia data, time-series data, text data, World Wide Web, etc.
- ***Classification according to the data model drawn on***: this classification categorizes data mining systems based on the data model involved such as relational database, object-oriented database, data warehouse, transactional, etc.
- ***Classification according to the king of knowledge discovered***: this classification categorizes data mining systems based on the kind of knowledge discovered or data mining functionalities, such as characterization, discrimination, association, classification, clustering, etc. Some systems tend to be comprehensive systems offering several data mining functionalities together.
- ***Classification according to mining techniques used***: Data mining systems employ and provide different techniques. This classification categorizes data mining systems according to the data analysis approach used such as machine learning, neural networks, genetic algorithms, statistics, visualization, database-oriented or data warehouse-oriented, etc. The classification can also take into account the degree of user interaction involved in the data mining process such as query-driven systems, interactive exploratory systems, or autonomous systems. A comprehensive system would provide a wide variety of data mining techniques to fit different situations and options, and offer different degrees of user interaction.

1.6 Data Mining Issues

Data mining algorithms embody techniques that have sometimes existed for many years, but have only lately been applied as reliable and scalable tools that time and again outperform older classical statistical methods. While data mining is still in its infancy, it is becoming a trend and ubiquitous. Before data

mining develops into a conventional, mature and trusted discipline, many still pending issues have to be addressed. Some of these issues are addressed below. Note that these issues are not exclusive and are not ordered in any way.

Mining methodology and user interaction issues:

These reflect the kinds of knowledge mined the ability to mine knowledge at multiple granularities, the use of domain knowledge, ad hoc mining, and knowledge visualization.

- Mining different kinds of knowledge databases: Data mining should cover a wide spectrum of data analysis and knowledge discovery tasks, including data characterization, discrimination, association, classification, clustering, tread and deviation analysis, and similarity analysis.
- Interactive mining of knowledge at multiple levels of abstraction: The data mining process should be interactive.
- Interactive mining allows users to focus the search for patterns, providing and refining data mining requests based on returned results.
- Incorporation of background knowledge: Background knowledge may be used to guide the discovery process and allow discovered patterns to be expressed in concise terms and at different levels of abstraction.
- Data mining query languages and ad hoc mining: Relational query languages (such as SQL) allow users to pose ad hoc queries for data retrieval.
- Presentation and visualization of data mining results: Discovered knowledge should be expressed in high-level languages, visual representations, or other expressive forms so that knowledge can be easily understood and directly usable by humans.
- Handling noisy or incomplete data: When mining data regularities, these objects may confuse the process, causing the knowledge model constructed to over fit the data.
- Pattern evaluation--the interestingness problem: A data mining system can uncover thousands of patterns. Many of the patterns discovered may be uninteresting to the given user, representing common knowledge or lacking novelty.

Performance issues:

These include efficiency, scalability, and parallelization of data mining algorithms.

- Efficiency and scalability of data mining algorithms: To effectively extract information from a huge amount of data in databases, data mining algorithms must be efficient and scalable.
- Parallel, distributed, and incremental mining algorithms: The huge size of many databases, the wide distribution of data, and the computational complexity of some data mining methods are factors motivating the development of algorithms that divide data into partitions that can be processed in parallel.

Issues relating to the diversity of database types:

- Handling of relational and complex types of data: Specific data mining systems should be constructed for mining specific kinds of data.
- Mining information from heterogeneous databases and global information systems: Local- and wide-area computer networks (such as the Internet) connect many sources of data, forming huge, distributed, and heterogeneous databases.

The above issues are considered major requirements and challenges for the further evolution of data mining technology. Some of the challenges have been addressed in recent data mining research and development, to a certain extent, and are now considered requirements, while others are still in the research stage.

I.7 Applications of Data Mining

Data Mining has many and varied fields of application some of which are listed below.

→ Sales/Marketing

- Identify buying patterns from customers
- Find association among customer demographic characteristics
- Predict response to mailing campaigns
- Market basket analysis

→ Banking

- Credit card fraudulent detection
- Identity 'loyal' customers
- Predict customers likely to change their credit card affiliation
- Determine credit card spending by customer groups
- Find hidden correlation's between different financial indicators
- Identify stock trading rules from historical market data

→ Insurance and Health care

- Claims analysis i.e. which medical procedures are claimed together
- Predict which customers will buy new policies
- Identify behavior patterns of risky customers
- Identify fraudulent behavior

→ Transportation

- Determine the distribution schedules among outlets
- Analyze loading patterns

→ Medicine

- Characterize patient behavior
- Identify successful medical therapies for different illnesses
- Classifying gene structures

→ Production quality control

- Determine those combinations of production factors that influence the quality of the end-product
- Allows the process engineers to explain why certain products fail the final test to increase the quality of the production process

I.8 : Data warehouse and OLAP technology for Data mining

The construction of data warehouses, which involves data cleaning and data integration, can be viewed as an important preprocessing step for data mining. Moreover, data warehouses provide on-line analytical processing (OLAP) tools for the interactive analysis of multidimensional data of varied granularities, which facilitates effective data mining. Furthermore, many other data mining functions such as classification, prediction, association, and clustering, can be integrated with OLAP operations to enhance interactive mining of knowledge at multiple levels of abstraction. Hence, data warehouse

has become an increasingly important platform for data analysis and online analytical processing and will provide an effective platform for data mining.

1.8.1 What is a data warehouse?

Data warehousing provides architectures and tools for business executives to systematically organize, understand, and use their data to make strategic decisions. A large number of organizations have found that data warehouse systems are valuable tools in today's competitive, fast evolving world. In the last several years, many firms have spent millions of dollars in building enterprise-wide data warehouses. Many people feel that with competition mounting in every industry, data warehousing is the latest must-have marketing weapon | a way to keep customers by learning more about their needs.

Data warehouses have been defined in many ways, making it difficult to formulate a rigorous definition. Loosely speaking, a data warehouse refers to a database that is maintained separately from an organization's operational databases. Data warehouse systems allow for the integration of a variety of application systems. They support information processing by providing a solid platform of consolidated, historical data for analysis.

According to W. H. Inmon, a leading architect in the construction of data warehouse systems, a data warehouse is a subject-oriented, integrated, time-variant, and nonvolatile collection of data in support of management's decision making process." This short, but comprehensive definition presents the major features of a data warehouse. The four keywords,

→ subject-oriented
→ integrated
→ time-variant
→ nonvolatile

Distinguish data warehouses from other data repository systems, such as relational database systems, transaction processing systems, and file systems.

Subject-oriented: A data warehouse is organized around major subjects., such as customer, vendor, product, and sales. Rather than concentrating on the day-to-day operations and transaction processing of an organization, a data

warehouse focuses on the modeling and analysis of data for decision makers. Hence, data warehouses typically provide a simple and concise view around particular subject issues by excluding data that are not useful in the decision support process.

Integrated: A data warehouse is usually constructed by integrating multiple heterogeneous sources, such as relational databases, flat files, and on-line transaction records. Data cleaning and data integration techniques are applied to ensure consistency in naming conventions, encoding structures, attribute measures, and so on.

Time-variant: Data are stored to provide information from a historical perspective (e.g., the past 5-10 years). Every key structure in the data warehouse contains, either implicitly or explicitly, an element of time.

Nonvolatile: A data warehouse is always a physically separate store of data transformed from the application data found in the operational environment. Due to this separation, a data warehouse does not require transaction processing, recovery, and concurrency control mechanisms. It usually requires only two operations in data accessing: initial loading of data and access of data.

In sum, a data warehouse is a semantically consistent data store that serves as a physical implementation of a decision support data model and stores the information on which an enterprise needs to make strategic decisions. A data warehouse is also often viewed as architecture, constructed by integrating data from multiple heterogeneous sources to support structured and/or ad hoc queries, analytical reporting, and decision making.

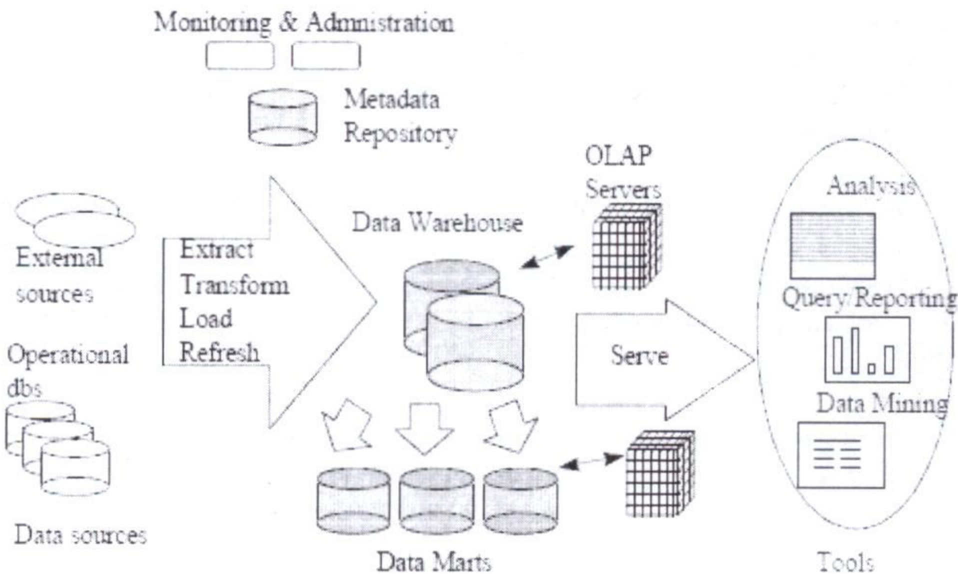


Figure I.8.1. Data Warehousing Architecture

The data warehousing architecture includes tools for extracting data from multiple operational databases and external sources; for cleaning, transforming and integrating this data; for loading data into the data warehouse; and for periodically refreshing the warehouse to reflect updates at the sources and to purge data from the warehouse, perhaps onto slower archival storage. In addition to the main warehouse, there may be several departmental data marts. Data in the warehouse and data marts is stored and managed by one or more warehouse servers, which present multidimensional views of data to a variety of front end tools: query tools, report writers, analysis tools, and data mining tools. Finally, there is a repository for storing and managing metadata, and tools for monitoring and administering the warehousing system.

Based on the above, data warehousing can be viewed as the process of constructing and using data warehouses. The construction of a data warehouse requires data integration, data cleaning, and data consolidation. The utilization of a data warehouse often necessitates a collection of decision support

technologies. This allows **knowledge workers** (e.g., managers, analysts, and executives) to use the warehouse to quickly and conveniently obtain an overview of the data, and to make sound decisions based on information in the warehouse.

1.8.2 Organizations and data warehouses

Many organizations are using this information to support business decision making activities, including

1. Increasing customer focus, which includes the analysis of customer buying patterns (such as buying preference, buying time, budget cycles, and appetites for spending),
2. Repositioning products and managing product portfolios by comparing the performance of sales by quarter, by year, and by geographic regions, in order to fine-tune production strategies,
3. Analyzing operations and looking for sources of profit, and
4. Managing the customer relationships, making environmental corrections, and managing the cost of corporate assets.

Data warehousing is also very useful from the point of view of heterogeneous database integration. Many organizations typically collect diverse kinds of data and maintain large databases from multiple, heterogeneous, autonomous, and distributed information sources. To integrate such data, and provide easy and efficient access to it is highly desirable, yet challenging. Much effort has been spent in the database industry and research community towards achieving this goal.

The traditional database approach to heterogeneous database integration is to build wrappers and integrators (or mediators) on top of multiple, heterogeneous databases. A variety of data joiner (DB2) and data blade (Informix) products belong to this category. When a query is posed to a client site, Meta data dictionary is used to translate the query into queries appropriate for the individual heterogeneous sites involved. These queries are then mapped and sent to local query processors.

The results returned from the different sites are integrated into a global answer set. This query-driven approach requires complex information filtering and integration processes, and competes for resources with processing at local

sources. It is inefficient and potentially expensive for frequent queries, especially for queries requiring aggregations.

Data warehousing provides an interesting alternative to the traditional approach of heterogeneous database integration described above. Rather than using a query-driven approach, data warehousing employs an update-driven approach in which information from multiple, heterogeneous sources is integrated in advance and stored in a warehouse for direct querying and analysis. Unlike on-line transaction processing databases, data warehouses do not contain the most current information.

However, a data warehouse brings high performance to the integrated heterogeneous database system since data are copied, preprocessed, integrated, annotated, summarized, and restructured into one semantic data store. Furthermore, query processing in data warehouses does not interfere with the processing at local sources. Moreover, data warehouses can store and integrate historical information and support complex multidimensional queries. As a result, data warehousing has become very popular in industry.

1.8.3 OLTP and OLAP

Since most people are familiar with commercial relational database systems, it is easy to understand what a data warehouse is by comparing these two kinds of systems.

- ***On-line transaction processing (OLTP) systems***
- ***On-line analytical processing (OLAP) systems***

The major task of on-line operational database systems is to perform on-line transaction and query processing. These systems are called ***on-line transaction processing (OLTP) systems***. They cover most of the day-today operations of an organization, such as, purchasing, inventory, manufacturing, banking, payroll, registration, and accounting.

Data warehouse systems, on the other hand, serve users or knowledge workers in the role of data analysis and decision making. Such systems can organize and present data in various formats in order to accommodate the diverse needs of the different users. These systems are known as ***on-line analytical processing (OLAP) systems***.

The major distinguishing features between OLTP and OLAP are summarized as follows.

Users and system orientation:

- An OLTP system is customer-oriented and is used for transaction and query processing by clerks, clients, and information technology professionals.
- An OLAP system is market-oriented and is used for data analysis by knowledge workers, including managers, executives, and analysts.

Data contents:

- An OLTP system manages current data that, typically, are too detailed to be easily used for decision making.
- An OLAP system manages large amounts of historical data, provides facilities for summarization and aggregation, and stores and manages information at different levels of granularity. These features make the data easier for use in informed decision making.

Database design:

- An OLTP system usually adopts an entity-relationship (ER) data model and an application oriented database design.
- An OLAP system typically adopts either a star or snow flake model and a subject-oriented database design.

View:

- An OLTP system focuses mainly on the current data within an enterprise or department, without referring to historical data or data in different organizations.
- In contrast, an OLAP system often spans multiple versions of a database schema, due to the evolutionary process of an organization. OLAP systems also deal with information that originates from different organizations, integrating information from many data stores. Because of their huge volume, OLAP data are stored on multiple storage media.

Access patterns:

- The access patterns of an OLTP system consist mainly of short, atomic transactions. Such a system requires concurrency control and recovery mechanisms.

- However, accesses to OLAP systems are mostly read-only operations (since most data warehouses store historical rather than up-to-date information), although many could be complex queries.

Other features which distinguish between OLTP and OLAP systems include database size, frequency of operations, and performance metrics. These are summarized in Table I.8.3.1

Table I.8.3.1: Comparison between OLTP and OLAP systems

Feature	OLTP	OLAP
Characteristic	operational processing	informational processing
Orientation	transaction	analysis
User	clerk, DBA, database professional	knowledge worker (e.g., manager, executive, analyst)
Function	day-to-day operations	long term informational requirements, decision support
DB design	E-R based, application-oriented	star/snow flake, subject-oriented
Data	current; guaranteed up-to-date	historical; accuracy maintained over time
Summarization	primitive, highly detailed	summarized, consolidated
View	detailed, flat relational	summarized, multidimensional
Unit of work	short, simple transaction	complex query
Access	read/write	mostly read
Focus	data in	information out
Operations	index/hash on primary key	lots of scans
# of records accessed	tens	millions
# of users	thousands	hundreds
DB size	100 MB to GB	100 GB to TB
Priority	high performance, high availability	high flexibility, end-user autonomy
Metric	transaction throughput	query throughput, response time

Finally, the separation of operational databases from data warehouses is based on the different structures, contents, and uses of the data in these two systems. Decision support requires historical data, whereas operational databases do not typically maintain historical data. In this context, the data in operational databases, though abundant, is usually far from complete for decision making. Decision support requires consolidation (such as aggregation and summarization) of data from heterogeneous sources, resulting in high quality, cleansed and integrated data.

In contrast, operational databases contain only detailed raw data, such as transactions, which need to be consolidated before analysis. Since the two systems provide quite different functionalities and require different kinds of data, it is presently necessary to maintain separate databases. However, many vendors of operational relational database management systems are beginning to optimize such systems so as to support OLAP queries. As this trend continues, the separation between operational DBMSs and OLAP systems is expected to decrease.

I.9 : Optimizing Data for Mining

Today's real-world databases are highly susceptible to noise, missing, and inconsistent data due to their typically huge size, often several gigabytes or more.

How can the data be preprocessed in order to help improve the quality of the data, and consequently, of the mining results?

How can the data be preprocessed so as to improve the efficiency and ease of the mining process?

There are a number of data preprocessing techniques. They are

- Data cleaning
- Data integration
- Data transformations
- Data reduction

Data cleaning can be applied to remove noise and correct inconsistencies in the data. ***Data integration*** merges data from multiple sources into a coherent data store, such as a data warehouse or a data cube. ***Data transformations***, such

as normalization, may be applied. For example, normalization may improve the accuracy and efficiency of mining algorithms involving distance measurements. ***Data reduction*** can reduce the data size by aggregating, eliminating redundant features, or clustering, for instance. These data processing techniques, when applied prior to mining, can substantially improve the overall quality of the patterns mined and/or the time required for the actual mining.

1.9.1 Data Preprocessing

Imagine that you are a manager at "***AllElectronics***" and have been charged with analyzing the company's data with respect to the sales at your branch. You immediately set out to perform this task.

- → You carefully inspect the company's database or data warehouse, identifying and selecting the attributes or dimensions to be included in your analysis, such as item, price, and units sold.
- → You note that several of the attributes for various tuples have no recorded value.
- → For your analysis, you would like to include information as to whether each item purchased was advertised as on sale, yet you discover that this information has not been recorded.
- → Furthermore, users of your database system have reported errors, unusual values, and inconsistencies in the data recorded for some transactions.

In other words, the ***data you wish to analyze by data mining techniques are incomplete*** (lacking attribute values or certain attributes of interest, or containing only aggregate data), ***noisy*** (containing errors, or outlier values which deviate from the expected), and ***inconsistent*** (e.g., containing discrepancies in the department codes used to categorize items). Incomplete, noisy, and inconsistent data are commonplace properties of large, real-world databases and data warehouses.

Incomplete data can occur for a number of reasons. *Attributes of interest may not always be available*, such as customer information for sales transaction data. Other *data may not be included* simply because it was not considered important at the time of entry. *Relevant data may not be recorded* due to a misunderstanding, or because of equipment malfunctions. Data that were

inconsistent with other recorded *data may have been deleted*. Furthermore, the recording of the history or modifications to the *data may have been overlooked*. Missing data, particularly for tuples with missing values for some attributes, may need to be inferred. ***Data can be noisy***, having *incorrect attribute values*, owing to the following. The data collection instruments used may be faulty. There may have been *human or computer errors* occurring at data entry. *Errors in data transmission* can also occur. There *may be technology limitations*, such as limited buffer size for coordinating synchronized data transfer and consumption. Incorrect data may also result *from inconsistencies in naming conventions* or data codes used. Duplicate tuples also require data cleaning.

Data cleaning routines work to "clean" the data by filling in missing values, smoothing noisy data, identifying or removing outliers, and resolving inconsistencies. Dirty data can cause confusion for the mining procedure. Although most mining routines have some procedures for dealing with incomplete or noisy data, they are not always robust. Instead, they may concentrate on avoiding over fitting the data to the function being modeled. Therefore, a useful preprocessing step is to run your data through some data cleaning routines.

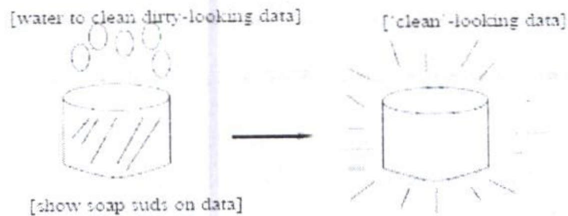
Getting back to your task at AllElectronics, suppose that you would like to include data from multiple sources in your analysis. This would involve integrating multiple databases, data cubes, or files, i.e., ***data integration***. Yet some attributes representing a *given concept may have different names in different databases*, causing inconsistencies and redundancies. For example, the attribute for customer identification may be referred to as customer id is one data store, and cust_id in another. *Naming inconsistencies* may also occur for attribute values. For example, the same first name could be registered as "Bill" in one database, but "William" in another, and "B" in the third. Furthermore, you suspect that some attributes may be derived or inferred from others (e.g., annual revenue). Having a large amount of redundant data may slow down or confuse the knowledge discovery process. Clearly, in addition to data cleaning, *steps must be taken to help avoid redundancies during data integration*. Typically, data cleaning and data integration are performed as a preprocessing step when preparing the data for a data warehouse. Additional data cleaning may

be performed to detect and remove redundancies that may have resulted from data integration.

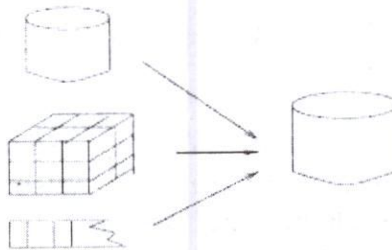
Getting back to your data, you have decided, say, that you would like to use a distance-based mining algorithm for your analysis, such as neural networks, nearest neighbor classifiers, or clustering. Such methods provide better results if the data to be analyzed have been normalized, that is, scaled to a specific range such as [0, 1.0]. Your customer data, for example, contains the attributes age, and annual salary. The annual salary attribute can take many more values than age. Therefore, *if the attributes are left un-normalized*, then distance measurements taken on annual salary will generally outweigh distance measurements taken on age. Furthermore, it would be useful for your analysis to *obtain aggregate information* as to the sales per customer region | something which is not part of any precomputed data cube in your data warehouse. You soon realize that ***data transformation*** operations, such as *normalization and aggregation, are additional data preprocessing procedures* that would contribute towards the success of the mining process.

"Hmmm", you wonder, as you consider your data even further. "The data set I have selected for analysis is huge ; it is sure to slow or wear down the mining process. Is there any way I can `reduce' the size of my data set, without jeopardizing the data mining results?" ***Data reduction*** obtains a reduced representation of the *data set that is much smaller in volume, yet produces the same (or almost the same) analytical results*. There are a number of strategies for data reduction. These include ***data aggregation*** (e.g., building a data cube), ***dimension reduction*** (e.g., removing irrelevant attributes through correlation analysis), ***data compression*** (e.g., using encoding schemes such as minimum length encoding or wavelets), and ***numerosity reduction*** (e.g., "replacing" the data by alternative, smaller representations such as clusters, or parametric models). Data can also be "reduced" by generalization, where low level concepts such as city for customer location, are replaced with higher level concepts, such as region or province or state. A concept hierarchy is used to organize the concepts into varying levels of abstraction. ***Data discretization*** is a form of data reduction that is very useful for the automatic generation of concept hierarchies from numerical data.

Data Cleaning



Data Integration



Data Transformation -2, 32, 100, 59, 48 → -0.02, 0.32, 1.00, 0.59, 0.48

Data Reduction

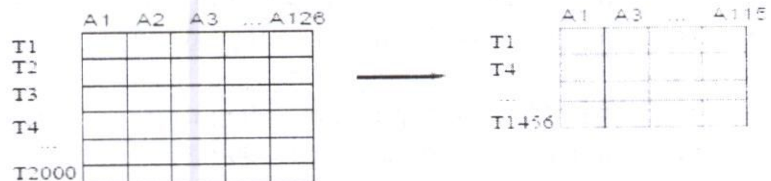


Figure I.9.1 : Forms of data preprocessing

Figure I.9.1 summarizes the data preprocessing steps described here. Note that the above categorization is not mutually exclusive. For example, the removal of redundant data may be seen as a form of data cleaning, as well as data reduction.

In summary, real-world data tend to be dirty, incomplete, and inconsistent. Data preprocessing techniques can improve the quality of the data, thereby helping to improve the accuracy and efficiency of the subsequent mining process. Data preprocessing is therefore an important step in the knowledge discovery process, since quality decisions must be based on quality data. Detecting data anomalies, rectifying them early, and reducing the data to be analyzed can lead to huge payoffs for decision making.

1.10 Summary

"***Data warehouse***" is a repository of multiple heterogeneous data sources. organized under a unified schema at a single site in order to facilitate management decision making.

Data warehouse technology includes data cleansing, data integration and On-Line Analytical Processing (**OLAP**).

The abundance of data, coupled with the need for powerful data analysis tools has been described as a *data rich but information poor* situation. As a result, data collected in large databases and become "***data tombs***" – which are nothing but data archives.

Data mining is

- Extracting or "mining" knowledge from large amounts of data.
- Data – driven discovery and modeling of hidden patterns in large volumes of data.
- Extraction of implicit, previously unknown and unexpected, potentially, extremely useful information from data.

"*Extraction of interesting information or patterns from data in large databases is known as* ***data mining***."

Data Mining refers to "*using a variety of techniques to identify nuggets of information or decision-making knowledge in bodies of data and extracting these in such a way that they can be put to use in the areas such as decision support, prediction, forecasting and estimation. The data is often voluminous, but as it stands of low value as no direct use can be made of it; it is the hidden information in the data that is useful*"

We have been collecting a myriad of data, from simple numerical measurements and text documents, to more complex information such as spatial data, multimedia channels, and hypertext documents.

- Business transactions:
- Scientific data:
- Medical and personal data:
- Surveillance video and pictures:
- Satellite sensing:
- Games:
- Digital media:
- CAD and Software engineering data:
- Virtual Worlds:
- Text reports and memos (e-mail messages):
- The World Wide Web repositories:

The Knowledge Discovery in Databases process comprises of a few steps leading from raw data collections to some form of new knowledge. The iterative process consists of the following steps:

- Data cleaning
- Data integration
- Data selection
- Data transformation
- Data mining
- Pattern evaluation
- Knowledge representation

***Data cleaning*:** also known as data cleansing, it is a phase in which noise data and irrelevant data are removed from the collection.
***Data integration*:** at this stage, multiple data sources, often heterogeneous, may be combined in a common source.
***Data selection*:** at this step, the data relevant to the analysis is decided on and retrieved from the data collection.
***Data transformation*:** also known as data consolidation, it is a phase in which the selected data is transformed into forms appropriate for the mining procedure.
***Data mining*:** it is the crucial step in which clever techniques are applied to extract patterns potentially useful.
***Pattern evaluation*:** in this step, strictly interesting patterns representing knowledge are identified based on given measures.

***Knowledge representation*:** is the final phase in which the discovered knowledge is visually represented to the user. This essential step uses visualization techniques to help users understand and interpret the data mining results.

Data mining is being put into use and studied for

- Flat files
- Relational databases
- Data Warehouses
- Transaction Databases
- Multimedia Databases
- Spatial Databases
- Time-Series Databases
- World Wide Web

Data Mining Functionalities

The data mining functionalities and the variety of knowledge they discover are briefly presented in the following list:

- Characterization
- Discrimination:
- Association analysis:
- Classification:
- Prediction:
- Clustering:

Some of the Data mining issues are listed below.

- *Mining methodology and user interaction issues:*
- *Performance issues:*
- *Issues relating to the diversity of database types:*

Data Mining has many and varied fields of application some of which are listed below.

→ Sales/Marketing
→ Banking
→ Insurance and Health care
→ Transportation
→ Medicine
→ Production quality control

Data warehouse systems, on the other hand, serve users or knowledge workers in the role of data analysis and decision making. Such systems can organize and present data in various formats in order to accommodate the diverse needs of the different users. These systems are known as *on-line analytical processing (OLAP) systems*.

There are a number of data preprocessing techniques. They are

- Data cleaning
- Data integration
- Data transformations
- Data reduction

Data cleaning can be applied to remove noise and correct inconsistencies in the data.

Data integration merges data from multiple sources into a coherent data store, such as a data warehouse or a data cube.

Data transformations, such as normalization, may be applied. For example, normalization may improve the accuracy and efficiency of mining algorithms involving distance measurements.

Data reduction can reduce the data size by aggregating, eliminating redundant features, or clustering, for instance.

UNIT II DATA MINING PRIMITIVES, ASSOCIATION RULES, CLASSIFICATION & PREDICTION

Structure

II.0 Objectives

In this unit we are going to discuss about the data mining primitives and some of the techniques used in data mining. This unit also elaborates in detail about the association rules, fuzzy sets and logic, classification and prediction. It also discusses the estimation errors related to Data mining.

At the end of this unit we can

- Understand the data mining primitives
- Got in depth knowledge about association rule mining and fuzzy sets and logic
- Gain knowledge about the classification and prediction
- Elaborate the estimation errors of data mining

II.1 : Data Mining Primitives

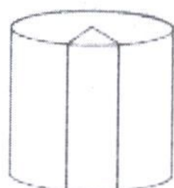
Each user will have a data mining task in mind, that is, some form of data analysis that he or she would like to have performed. ***A data mining task can be specified in the form of a data mining query, which is input to the data mining***

system. A data mining query is defined in terms of data mining task primitives. These primitives allow the user to inter-actively communicate with the data mining system during discovery in order to direct the mining process. or examine the findings from different angles or depths. The data mining primitives specify the following. as illustrated in Figure II.1.1.

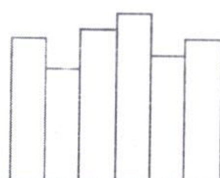
The set of task-relevant data to be mined: This specifies the portions of the database or the set of data in which the user is interested. This includes the database attributes or data warehouse dimensions of interest (referred to as the relevant attributes or dimensions).

The kind of knowledge to be mined: This specifies the data mining functions to be performed, such as characterization. discrimination. association or correlation analysis. classification, prediction, clustering. outlier analysis. or evolution analysis.

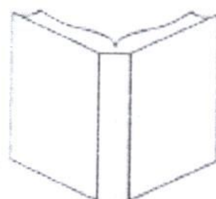
The background knowledge to be used in the discovery process: This knowledge about the domain to be mined is useful for guiding the knowledge discovery process and for evaluating the patterns found. Concept hierarchies are a popular form of background knowledge, which allow data to be mined at multiple levels of abstraction. An example of a concept hierarchy for the attribute (or dimension) age is shown in Figure 1.14. User beliefs regarding relationships in the data are another form of back ground knowledge.



Task-relevant data
Database or data warehouse name
Database tables or data warehouse cubes
Conditions for data selection
Relevant attributes or dimensions
Data grouping criteria



Knowledge type to be mined
Characterization
Discrimination
Association/correlation
Classification/prediction
Clustering



Background knowledge
Concept hierarchies
User beliefs about relationships in the data



Pattern interestingness measures
Simplicity
Certainty (e.g., confidence)
Utility (e.g., support)
Novelty



Visualization of discovered patterns
Rules, tables, reports, charts, graphs, decision trees, and cubes
Drill-down and roll-up

Figure II.1.1: Primitives for specifying a data mining task

The interestingness measures and thresholds for pattern evaluation: They may be used to guide the mining process or, after discovery, to evaluate the discovered patterns. Different kinds of knowledge may have different interestingness measures. For example, interestingness measures for association rules include support and confidence. Rules whose support and confidence values are below user-specified thresholds are considered uninteresting.

The expected representation for visualizing the discovered patterns: This refers to the form in which discovered patterns are to be displayed, which may include rules, tables, charts, graphs, decision trees, and cubes.

A data mining query language can be designed to incorporate these primitives, allowing users to flexibly interact with data mining systems. Having a data mining query language provides a foundation on which user-friendly graphical interfaces can be built. This facilitates a data mining system's communication with other information systems and its integration with the overall information processing environment.

Designing a comprehensive data mining language is challenging because data mining covers a wide spectrum of tasks, from data characterization to evolution analysis. Each task has different requirements. The design of an effective data mining query language requires a deep understanding of the power, limitation, and underlying mechanisms of the various kinds of data mining tasks.

There are several proposals on data mining languages and standards. In this book, we use a data mining query language known as DMQL (Data Mining Query Language), which was designed as a teaching tool, based on the above primitives.

Example:

Suppose, as a marketing manager of ***AllElectronics***, you would like to classify customers based on their buying patterns. You are especially interested in those customers whose salary is no less than $40,000, and who have bought more than $1,000 worth of items, each of which is priced at no less than $100. In

SL - 19 : 4

particular, you are interested in the customer's age, income, the types of items purchased, the purchase location, and where the items were made. You would like to view the resulting classification in the form of rules. This data mining query is expressed in DMQL3 as follows, where each line of the query has been enumerated to aid in our discussion.

(1) use database AllElectronics db
(2) use hierarchy location hierarchy for T.branch, age hierarchy for C.age
(3) mine classification as promising customers
(4) in relevance to C.age, C.income, I.type, I.place made, T.branch
(5) from customer C, item I, transaction T
(6) where I.item ID = T.item ID and C.cust ID = T.cust ID and C.income , 40.000 and I.price , 100
(7) group by T.cust ID

11.2 : Association Rules

11.2.1 Introduction

Association rule mining finds interesting association or correlation relationships among a large set of data items. With massive amounts of data continuously being collected and stored, many industries are becoming interested in mining association huge amounts of business transaction records can help in many business decision making processes, such as catalog design, cross-marketing, and loss-leader analysis.

A typical example of association rule mining is market basket analysis. This process analyzes customer buying habits by finding associations between the different items that customers place in their "shopping baskets". The discovery of such associations can help retailers develop marketing strategies by gaining insight into which items are frequently purchased together by customers. For instance, if customers are buying milk, how likely are they to also buy bread (and what kind of bread) on the same trip to the supermarket? Such information

can lead to increased sales by helping retailers do selective marketing and plan their shelf space. For example, placing milk and bread within close proximity may further encourage the sale of these items together within single visits to the store.

How can we find association rules from large amounts of data, where the data are either transactional or relational? Which association rules are the most interesting? How can we help or guide the mining procedure to discover interesting associations? What language constructs is useful in defining a data mining query language for association rule mining? The following section will answer the above mentioned questions.

II.2.2 Association Rule mining

Association rule mining searches for interesting relationships among items in a given data set. This section provides an introduction to association rule mining. We begin by presenting an example of market basket analysis, the earliest form of association rule mining. The basic concepts of mining associations are given and we present a road map to the different kinds of association rules that can be mined.

Market Basket Analysis: A Motivating example for Association Rule Mining

Suppose, as manager of an ***ABCompany*** branch, you would like to learn more about the buying habits of your customers. Specifically, you wonder, "Which groups or Sets of items are customers likely to purchase on a given trip to the store?" To answer your question, market basket analysis may be performed on the retail data of customer Transactions at your store. The results may be used to plan marketing or advertising Strategies, as well as catalog design. For instance, market basket analysis may help Managers design different store layouts.

In one strategy, items that are frequently purchased together can be placed in close proximity in order to further encourage the Sale of such items together. If customers who purchase computers also tend to buy financial management

software at the same time, then placing the hardware display close to the software display may help to increase the sales of both these items.

In an Alternative strategy, placing hardware and software at opposite ends of the store may entice customers who purchase such items to pick up other items along the way. For Instance, after deciding on an expensive computer, a customer may observe security Systems for sale while heading towards the software display to purchase financial Management software and may decide to purchase a home security system as well.

Market basket analysis can also help retailers to plan which items to put on sale at reduced prices. If customers tend to purchase computers and printers together, then having a sale on printers may encourage the sale of printers as well as computers. If we think of the universe as the set of items available at the store, then each item has a Boolean variable representing the presence or absence of that item. A Boolean Vector of values assigned to these variables can then represent each basket. The Boolean Vectors can be analyzed for buying patterns that reflect items that are frequently Associated or purchased together. These patterns can be represented in the form of Association rules.

For example, the information that customers who purchase computers also tend to buy financial management software at the same time is represented in Associated Rule below:

Computer=> financial_management_software
[support = 2%,confidence == 60%]

Rule ***support and confidence*** are two measures of rule interesting that were described earlier. They respectively reflect the usefulness and certainty of discovered rules. A support of 2% for Association Rule means that 2% of all the transaction under analysis show that computer and financial management software are purchased a computer also bought the software. Typically, association rules are considered interesting if they satisfy both a minimum support thresholds and a minimum confidence threshold. Users or domain experts can set such thersholds.

Another Example for Market Basket analysis

Let's start with Association mining with market basket data as the example. An itemset is the group of items. A k-itemset indicates the no. of items under study is K numbers. As part of a transaction (purchase by customer) one or more items from the itemset may be included. The occurrence/purchase of an item is indicated by a value 1 while non-inclusion is indicated by a value 0. Hence a typical market basket data like the one below:

	Book	**Pen**	**Pencil**	**Eraser**	**Sharpener**	**Crayons**	**Maps**	**A4 sheets**
T1	1	1	0	1	0	1	0	1
T2	0	1	0	1	0	1	0	0
T3	1	0	0	1	0	1	0	0

Hence it means

T1 – {Book,Pen,Eraser,Crayons}

T2 – {Pen,Eraser,Crayons}

T3 – {Book,Eraer,Crayons}

In the above example, we call it an 8-itemset. If you see the above representation of market basket data, one may think there are few additional info which are missing like the quantity purchased, Amount involved in the transactions/purchase. Of course, the association analysis can be extended to involve such detail.

The application of Association rule mining algorithm results in the discovery of rules/patterns of the following form:

{Pencil} – > {Eraser}
{Book,Maps}- > {Pen}

What it simply says is "If a customer bought a pencil he is more likely to buy an eraser". Mathematically, it says "Purchase of Pencil implies purchase of eraser". Once a pattern is discovered, it can used/integrated into decision support system to form strategies based on the rule. In the above case, the company may use this rule to do cross-selling i.e place pencil and eraser as close to each other which increases the sales and hence profit. Just imagine, if a number of such strong rules are discovered in a jewellery shop it would result in tremendous value. Now let me answer what "strong" rule means?

Now that we have defined what a rule is. we are posed with two important questions. Are all rules discovered by my algorithm is really useful meaningful? How confident I am about the rule? To answer these questions. we use some mathematical measure to quantify the usefulness and confidence. Most common evaluation measures for a rule are support and confidence measures. There are other measures namely lift. interest factor. correlation.

A support measure answers the first question. the interestingness measure. It is represented in percentage. It defines how many of my transactions support this rule. If it is say 4/100 it means just 4 out of 100 transactions involve this rule. then probably this is uninteresting so we may choose to ignore it. Hence our Association rule mining algorithm sets some threshold/min value for the support to eliminate uninteresting rules and retain the interesting ones. An example of uninteresting rule could be {pen} - > {eraser}. this could be an uninteresting rule as pen and eraser might be purchased as a matter of chance. i.e it has lower support.

Now having answered the interestingness criteria. we are left with determining the confidence of the rule. A confidence measure quantifies the confidence as a ratio of no. of transaction holding this rule valid against the no. of transactions involving this rule. Higher the value. more reliable is the rule. A strong rule indicates a rule with higher confidence value.

Lets quickly jump into details of the algorithm. The Association rule mining is carried out using the famous Apriori Algorithm. We will also talk about the variations of this algorithm to apply it for continuous data and hierarchical data. Before that. let's formalize the definition of the association analysis problem:

"Given a set of transactions. the problem is to find all rules/patterns with support >= minsup and confidence >= minconf"

11.2.3 Basic Concepts

→ Let $t = \{i_1, i_2 .... i_m\}$ be a set of items.

→ Let ***D***, the task-relevant data, is a set of database transactions where each transaction T is a set of items such that $T \subset \tau$ each transaction is association with an identifier, called ***TID***.

→ Let ***A*** be a set of items. A transaction ***T*** is said to contain ***A*** if and only if $A \subseteq T$.

→ An association rule is an implication of the form A=>B. $A \subset t$, $B \subset t$ and $A \cap B = f$.

→ The rule A=> B holds in the transaction set ***D*** with support ***s***, where ***s*** is the percentage of transaction in ***D*** that contains $A \cup B$ (i.e both ***A*** and ***B***). This is taken to be the probability, $P(A \cup B)$.

→ The rule A=>B has confidence ***c*** in the transaction in the transaction set ***D*** if ***c*** is the percentage of transactions in ***D*** containing ***A*** that also contain ***B***. This is taken to be the conditional probability $P(B/A)$. That is

→ Support $(A{=}{>}B)=P(A \cup B)$

→ Confidence $(A{=}{>}B)=p(B/A)$

→ Rules that satisfy both a minimum support threshold ***(min_sup)*** and a minimum confidence threshold ***(min_conf)*** are called strong.

→ By convention, we write support and confidence value so as to occur between 0% and 100% rather than 0 to 1.0.

A set of items is referred to as an ***itemset***. An ***itemset*** that contains k item is a ***k-itemset***. The set {computer, financial_management_software} is a ***2-itemset***. The occurrence frequency of an ***itemset*** is the number of transaction that contain the ***itemset***. This is also known, simply, as the frequency, support count or count of the ***itemset***. An ***itemset*** satisfies minimum support if the occurrence frequency of the ***itemset*** is greater than or equal to the product of ***min_sup*** and the total number of transactions in ***D***. The number of transaction required for the ***itemset*** to satisfy minimum support is therefore referred to as the minimum support count. If an ***itemset*** satisfies minimum support, then it is a frequent ***itemset*** .The set of frequent ***K-itemsets*** is commonly denoted by ***LK***.

"How is association rules mined from large databases?" Association rule mining is a two-step process:

1. ***Find all frequent itemsets (Frequent itemset generation)***: By definition, each of these itemsets will occur at least as frequently as a pre-determined minimum support count.
2. ***Generate strong association rules from the frequent itemsets (Rule Generation)***: By definition, these rules must satisfy minimum support and minimum support and minimum confidence. Additional interestingness measures can be applied, if desired.

The second step is the easiest, of the two. The overall performance of mining association rules is determined by the first step.

II.2.4 Association Rule Mining : A Road map

Market basket analysis is just one form of association rule mining; in fact, there are many kinds of association rules. Association rules can be classified in various ways, based on the following criteria:

Based on the types of values handled in the rule: if a rule concerns associations between the presence and absence of items, it is a Boolean association rule. For example, the rule above is a Boolean association rule obtained from market basket analysis.

If a rule describes associations between quantitative items or attributes, then it is a quantitative association rule. In these rules, quantitative values for items or attributes are partitioned into intervals. The following rule is an example of a quantitative association rule, where X is a variable representing a customer:

age(X,"30........39")^income(X,"42K.....48")

implies buys(X, high resolution TV)

Note that the quantitative attributes, age and income, have been discretized.

Based on dimensions in the data:

If the items or attributes in an association rule reference only one dimension, then it is a single-dimensional association rule. Note that above rule could be rewritten as buys (X,"computer") implies buys (X,"financial_management_software"). The first rule above is a single dimensional association rule since it refers to only one dimension, buys. If a rule references two or more dimensions, such as the dimensions buys, time_of_transaction, and customer_category, then it is a multidimensional association rule.

Based on the levels of abstraction in the rule set:

Some method for association rule mining can find rules at differing levels of abstraction. For example, suppose that a set of association rule mined includes the following rules:

age(x,"30...39") buys(x,"laptop computer")
age(x,"30...39") buys{x,"computer")

In above rules the items bought are referenced at different levels of abstraction. (e.g.,"computer" is a higher-level abstraction of "laptop computer"). We refer to the rule set mined as consisting of multilevel association rules. If, instead, the rules within a given set do not reference items or attributes at different levels of abstraction, then the set contains single-level association rules.

Based on various extensions to association mining:

Association mining can be extended to correlation analysis, where the absence or presence of correlated items can be identified. It can also be extended to mining maxpatterns (i.e., maximal frequent patterns) and frequent closed itemssets. A maxpattern is a frequent pattern, ***p***, such that any proper sub pattern of ***p*** is not frequent. A frequent closed itemeset is a frequent closed itemset where an itemset ***c*** is closed if there exists no proper superset of ***c***, ***c'*** such that every transaction containing c also contains ***c'***. Maxpatterns and frequent closed itemset can be used to substantially reduce the number of frequent itemsets generated in mining.

II.2.5 Mining single-dimensional Association Rules from Transactional Databases:

In this section, you will learn methods for mining the simplest form of association rules-single dimensional, single-level, Boolean association rules, such as those discussed for market basket analysis later. We begin by presenting a priori, a basic algorithm for improved efficiency and scalability is presented. Then methods for mining association rules that, unlike a priori, do not involve the generation of "candidate" frequent itemsets. The last section describes how principles from priori can be applied to improve the efficiency of answering iceberg queries, which are common in market basket analysis.

The Apriori Algorithm:

A brute force approach is very expensive task. Hence the approach followed by apriori algorithm is to break up the requirement of computing support and confidence as a two separate tasks. In the first step, frequent itemsets are generated i.e those itemsets which holds the criteria of minimum support. In the second and final step, Rule generation is made possible by evaluation the confidence measure. Let's visualize the approach diagrammatically as shown below:

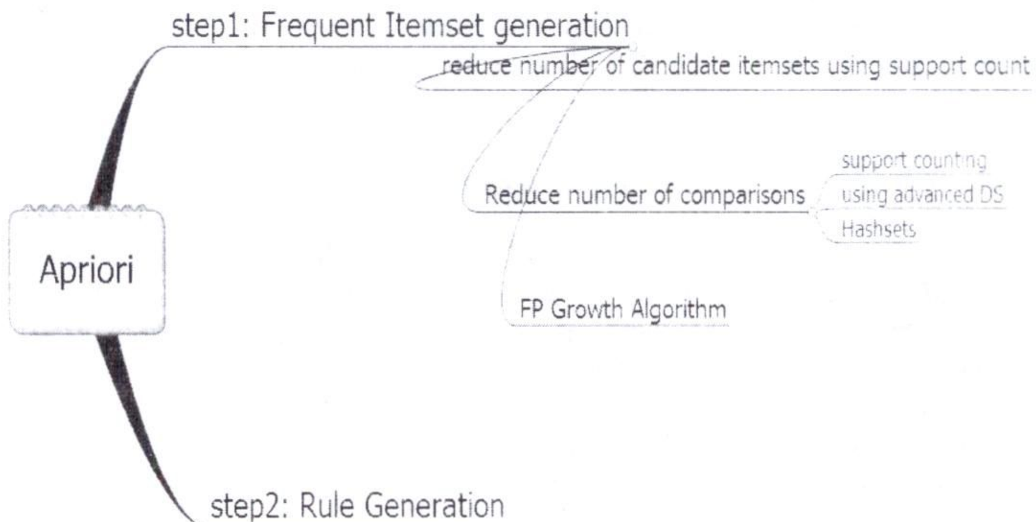


Figure II.2.2.1: Apriori Algorithm Steps

Measures could be classified into two categories – subjective and objective. A subjective measure often involves some heuristics and involves domain expertise to eliminate un-interesting rules while objective measure are domain independent measures. Support and confidence are good examples of objective measures. Objective measures could be either symmetric binary or asymmetric binary. The choice of measure depends on the type of application and it must be carefully chosen to get quality results.

Applications of apriori algorithms

The apriori algorithm can be extended to solving various other problems by making little modifications to the data representation methods. Data structures and algorithm.

1. To handle categorital and continous data. For example gender is categorical attribute and can be represented using two items namely gender='M' and gender = 'F'.
2. To handle concept of hierarchy in itemsets. For example, if IPod. Smartphone are two specific itemsets, then we can define a hierarchy item called electronic goods as a parent item.
3. To handle sequential pattern mining. Example: weblog mining, genome sequence mining, customer purchase behaviour.
4. Graph and sub-graph mining: eample – weblogs to identify navigation patterns, chemical structure analysis.
5. To identify infrequent patterns, negatively correlated patterns, etc.

The a priori Algorithm: Finding Frequent Itemsets Using Candidate Generation

A priori is an influential algorithm for mining frequent itemsets for Boolean association rules. The name of the algorithm is based on the fact that the algorithm uses prior knowledge of frequent itemset properties, as we shall see below. A priori employs a iterative approach known as a level-wise search, where f-itemsets are4 used to explore (k+1)-itemsets. First, the set of frequent 1-itemsets is found. This finding of each Lie requires one full scan of the database. To improve the efficiency of the level-wise generation of frequent itemsets, an important property called the a priori property, presented below, is used to reduce the search space. We will first describe this property, and then show an example illustrating its use.

In order to use the a priori property, all nonempty subsets of a frequent itemset must also be frequent. This property is based on the following observation. By definition, if an itemset (i.e.,I U A)cannot occur more frequently than I. Therefore, I U U is not frequent either, that is, P(I U A) < min_sup. This property belongs to a special category of properties called anti-monotone in the sense that if a set cannot pass a test, all of its supersets will fail the same test as well. It is called anti-monotone because the property is monotonic in the context of failing a test. To understand how the a priori property is used, let us look at how LK-1 is used to find LK.A two-step process is followed ,consisting of join and prune actions.

1. The join step:

→ To find LK, a set of candidate k-itemsets is generated by Joining Lk-1 with itself.

→ This set of candidates is denoted Ck. Let I1 and I2 be itemsets in Lk-1; The notation L1[j] refers to the jth item in li;(e.g.,l1[k-2]} refers to the second to the last item in li)

→ By convention, a priori method assumes that items within a transaction or itemset are sorted in kexicograpic order.

→ The join ,Lk-1 C Lk-1, is performed, where members of Lk-1 are joinable if their first (k-2) items are in comon.

→ That is, members l1 and l2 of Lk-1 are joined if (l1[1]=L2[1]^(l1[k-2]=L2[k-2]^(l1[k-1]=l2[k-2])

→ The conditional l1[k-1]<l2[k-2]., simply ensures that no duplicates are generated.

→ The resulting itemset formed by joining l1and l2 is l1[l1]l1[2]...l1[k-1]l2[k-1].

2. The prune step:

→ q is a superset of Lk that is,its members may or may not be frequent, but all of the frequent k-itemsets are included in ck.

→ A scan of the database to determine the count of each candidate in Ck would result in the determination of Lk (i.e., all candidates having a count no less than the maximum support count are frequent by definition, and therefore belong to Lk).

→ ck, however, can be huge, and so this could involve heavy computation. To reduce the size of Ck, the a priori property is used as follows. Any(k-1)-itemset that is not frequent cannot be a subset of a frequent k-itemset.

→ Hence, if any (k-1)-subset of a candidate k-itemset is not in Lk-1 then the candidate cannot be frequent either and so can be removed from Ck. This subset testing can be done quickly by maintaining a hash tree of all frequent itemsets.

procedure AprioriAlg()
begin
L_1 := {frequent 1-itemsets};
for (k := 2; $L_{k-1\ 0;\ k++\ )}$ **do** {
   C_k= apriori-gen(L_{k-1}); // new candidates
for all transactions t in the dataset **do** {
  for all candidates c C_k contained in t **do**
        c:count++
  }
  L_k = { c C_k | c:count >= min-support}
}
Answer := $_k$ L_k
end

It makes multiple passes over the database. In the first pass, the algorithm simply counts item occurrences to determine the frequent 1-itemsets (item sets with 1 item). A subsequent pass, say pass k, consists of two phases. First, the frequent item sets Lk-1 (the set of all frequent (k-1)-item sets) found in the (k-1)th pass are used to generate the candidate item sets Ck, using the apriori-gen() function. This function first joins Lk-1 with Lk-1, the joining condition being that the lexicographically ordered first k-2 items are the same. Next, it deletes all those item sets from the join result that have some (k-1)-subset that is not in Lk-1 yielding Ck.

The algorithm now scans the database. For each transaction, it determines which of the candidates in Ck are contained in the transaction using a hash-tree data structure and increments the count of those candidates. At the end of the

pass. Ck is examined to determine which of the candidates frequent, yielding Lk. The algorithm terminates when Lk becomes empty.

II.3 : Fuzzy sets and logic

II.3.1 Fuzzy sets

Fuzzy sets are sets whose elements have degrees of membership. Fuzzy sets were introduced simultaneously by Lotfi A. Zadeh and Dieter Klaua in 1965 as an extension of the classical notion of set. In classical set theory, the membership of elements in a set is assessed in binary terms according to a bivalent condition — an element either belongs or does not belong to the set. By contrast, fuzzy set theory permits the gradual assessment of the membership of elements in a set; this is described with the aid of a membership function valued in the real unit interval [0, 1]. Fuzzy sets generalize classical sets, since the indicator functions of classical sets are special cases of the membership functions of fuzzy sets, if the latter only take values 0 or 1. In fuzzy set theory, classical bivalent sets are usually called crisp sets. The fuzzy set theory can be used in a wide range of domains in which information is incomplete or imprecise, such as bioinformatics.

Fuzzy sets can be applied, for example, to the field of genealogical research. When an individual is searching in vital records such as birth records for possible ancestors, the researcher must contend with a number of issues that could be encapsulated in a membership function. Looking for an ancestor named John Henry Pittman, who you think was born in (probably eastern) Tennessee circa 1853 (based on statements of his age in later censuses, and a marriage record in Knoxville), what is the likelihood that a particular birth record for "John Pittman" is your John Pittman? What about a record in a different part of Tennessee for "J.H. Pittman" in 1851? (It has been suggested by Thayer Watkins that Zadeh's ethnicity is an example of a fuzzy set)

A fuzzy set is a pair (A,m) where A is a set and $m : A \rightarrow [0, 1]$.

For each $x \in A$, m(x) is called the grade of membership of x in (A,m). For a finite set A = {x1,...,xn}, the fuzzy set (A, m) is often denoted by {m(x1) / x1,...,m(xn) / xn}.

Let $x \in A$. Then x is called not included in the fuzzy set (A,m) if m(x) = 0, x is called fully included if m(x) = 1, and x is called a fuzzy member if 0 < m(x) < 1.[8] The set $\{x \in A \mid m(x) > 0\}$is called the support of (A, m) and the set $\{x \in A \mid m(x) = 1\}$is called its kernel.

Sometimes, more general variants of the notion of fuzzy set are used, with membership functions taking values in a (fixed or variable) algebra or structure L of a given kind; usually it is required that L be at least a poset or lattice. These are usually called L-fuzzy sets, to distinguish them from those valued over the unit interval. The usual membership functions with values in [0, 1] are then called [0, 1]-valued membership functions. These kinds of generalizations were first considered in 1967 by Joseph Goguen, who was a student of Zadeh.

II.3.2 Fuzzy logic

Fuzzy logic is a form of many-valued logic; it deals with reasoning that is approximate rather than fixed and exact. In contrast with traditional logic theory, where binary sets have two-valued logic: true or false, fuzzy logic variables may have a truth value that ranges in degree between 0 and 1. Fuzzy logic has been extended to handle the concept of partial truth, where the truth value may range between completely true and completely false. Furthermore, when linguistic variables are used, these degrees may be managed by specific functions.

Fuzzy logic began with the 1965 proposal of fuzzy set theory by Lotfi Zadeh. Though fuzzy logic has been applied to many fields, from control theory to artificiàl intelligence, it still remains controversial.

Degrees of truth

Fuzzy logic and probabilistic logic are mathematically similar – both have truth values ranging between 0 and 1 – but conceptually distinct, due to different interpretations. Fuzzy logic corresponds to "degrees of truth", while probabilistic logic corresponds to "probability, likelihood"; as these differ, fuzzy logic and probabilistic logic yield different models of the same real-world situations.

Both degrees of truth and probabilities range between 0 and 1 and hence may seem similar at first. For example, let a 100 ml glass contain 30 ml of water.

Then we may consider two concepts: Empty and Full. The meaning of each of them can be represented by a certain fuzzy set. Then one might define the glass as being 0.7 empty and 0.3 full. Note that the concept of emptiness would be subjective and thus would depend on the observer or designer. Another designer might equally well design a set membership function where the glass would be considered full for all values down to 50 ml. It is essential to realize that fuzzy logic uses truth degrees as a mathematical model of the vagueness phenomenon while probability is a mathematical model of ignorance. The same could be achieved using probabilistic methods, by defining a binary variable "full" that depends on a continuous variable that describes how full the glass is. There is no consensus on which method should be preferred in a specific situation.

Applying truth values

A basic application might characterize sub-ranges of a continuous variable. For instance, a temperature measurement for anti-lock brakes might have several separate membership functions defining particular temperature ranges needed to control the brakes properly. Each function maps the same temperature value to a truth value in the 0 to 1 range. These truth values can then be used to determine how the brakes should be controlled.

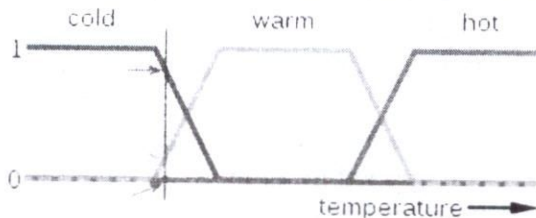


Figure : Fuzzy logic temperature

In this image, the meaning of the expressions cold, warm, and hot is represented by functions mapping a temperature scale. A point on that scale has three "truth values"—one for each of the three functions. The vertical line in the image represents a particular temperature that the three arrows (truth values) gauge. Since the red arrow points to zero, this temperature may be interpreted as "not hot". The orange arrow (pointing at 0.2) may describe it as "slightly warm" and the blue arrow (pointing at 0.8) "fairly cold".

Linguistic variables

While variables in mathematics usually take numerical values, in fuzzy logic applications, the non-numeric linguistic variables are often used to facilitate the expression of rules and facts.

A linguistic variable such as age may have a value such as young or its antonym old. However, the great utility of linguistic variables is that they can be modified via linguistic hedges applied to primary terms. The linguistic hedges can be associated with certain functions.

Example

Fuzzy set theory defines fuzzy operators on fuzzy sets. The problem in applying this is that the appropriate fuzzy operator may not be known. For this reason, fuzzy logic usually uses IF-THEN rules, or constructs that are equivalent, such as fuzzy associative matrices.

Rules are usually expressed in the form:

IF variable IS property THEN action

For example, a simple temperature regulator that uses a fan might look like this:

→ IF temperature IS very cold THEN stop fan
→ IF temperature IS cold THEN turn down fan
→ IF temperature IS normal THEN maintain level
→ IF temperature IS hot THEN speed up fan

There is no "ELSE" – all of the rules are evaluated, because the temperature might be "cold" and "normal" at the same time to different degrees.

The AND, OR, and NOT operators of Boolean logic exist in fuzzy logic, usually defined as the minimum, maximum, and complement; when they are defined this way, they are called the Zadeh operators. So for the fuzzy variables x and y:

NOT x = (1 - truth(x))

x AND y = minimum(truth(x), truth(y))

x OR y = maximum(truth(x), truth(y))

There are also other operators, more linguistic in nature, called hedges that can be applied. These are generally adverbs such as "very", or "somewhat", which modify the meaning of a set using a mathematical formula.

II.3.3 Fuzzy number

A fuzzy number is a convex, normalized fuzzy set $\tilde{A} \subseteq \mathbb{R}$ whose membership function is at least segmentally continuous and has the functional value μA(x) = 1 at precisely one element.

This can be likened to the funfair game "guess your weight," where someone guesses the contestant's weight, with closer guesses being more correct, and where the guesser "wins" if he or she guesses near enough to the contestant's weight, with the actual weight being completely correct (mapping to 1 by the membership function).

II.3.4 Fuzzy interval

A fuzzy interval is an uncertain set $\tilde{A} \subseteq \mathbb{R}$ with a mean interval whose elements possess the membership function value μA(x) = 1. As in fuzzy numbers, the membership function must be convex, normalized, atleast segmentally continuous.

II.3.5 Fuzzy relation equation

The fuzzy relation equation is an equation of the form A · R = B, where A and B are fuzzy sets, R is a fuzzy relation, and A · R stands for the composition of A with R.

II.4 : Classification and Prediction

II.4.1 Classification

Basically classification is a 2-step process; the first step is supervised learning for the sake of the predefined class label for training data set. Second step is classification accuracy evaluation. Likewise data prediction is also 2-step process.

II.4.1.1 Decision Trees

Decision trees are powerful and popular tool for classification and prediction. Decision trees represent rules. Decision tree is a classifier in the form of a tree structure where each node is either

- A leaf node, indicating a class of instances, or
- A decision node that specifies some test to be carried out on a single attribute value, which one branch and sub tree for each possible outcome of the test.

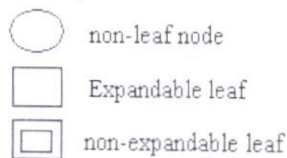
A decision tree can be used to classify and instance by starting at the rood of the tree and moving through it until a leaf node, which provides the classification of the instance.

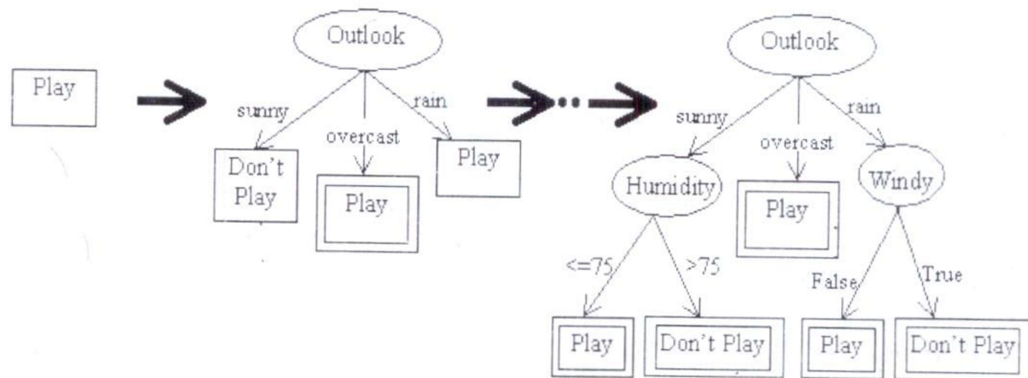
Constructing Decision Trees

Decision tree programs construct a decision tree T from a set of training cases. The original idea of construction of decision trees goes back to the work of Hoveland and Hunt on Concept Learning Systems (CLS) in the late 1950s.

The algorithm consists of five steps.

1. T ← the whole training set. Create a T node.
2. If all the samples in T are positive, create a "P" node with T as its parent and stop.
3. If all the samples in t are negative, create an "N" node with T as its parent and stop.
4. Select an attribute X with values v1, v2... vN and partition T into subsets T1, T2... TN according their values on X. Create N nodes Ti (i=1, 2,...., N) with T as their parent and X=vi as the label of the branch from T to Ti.
5. For each Ti do: T ← Ti and go to step 2.





(a)Initial Classification Tree (b)Intermediate Classification Tree (c) Final Classification Tree

Figure : Demonstration of Hunt's method : An example for a simple decision tree

Decision tree induction is a typical inductive approach to learn knowledge on classification. The key requirements to do mining with decision tree are:

→ Predefined classes: The categories to which cases are to be assigned must have been established beforehand (supervised data).

→ Discrete classes: A case does or does not belong to a particular class, and there must be for more cases than classes.

→ Sufficient data: Usually hundreds or even thousands of training cases.

→ "Logical" classification model: Classifier that can be only expressed as decision tree or set of production rules.

II.4.1.2. ID3 algorithm

The ID3 algorithm (Quinlan86) is a decision tree building algorithm which determines the classification of objects by testing the values of their properties. It builds the tree in a top down fashion, starting from a set of objects

and a specification of properties. At each node of the tree, a property is tested and the results used to partition the object set. This process is recursively done till the set in a given sub tree is homogeneous with respect to the classification criteria - in other words it contains objects belonging to the same category. This then becomes a leaf node. At each node, the property to test is chosen based on information theoretic criteria that seek to maximize information gain and minimize entropy. In simpler terms, that property is tested which divides the candidate set in the most homogeneous subsets.

II.4.1.3. C4.5 algorithm

This algorithm was proposed by Quinlan (1993). The C4.5 algorithm generates a classification-decision tree for the given data-set by recursive partitioning of data. The decision is grown using Depth-first strategy. The algorithm considers all the possible tests that can split the data set and selects a test that gives the best information gain. For each discrete attribute, one test with outcomes as many as the number of distinct values of the attribute is considered. For each continuous attribute, binary tests involving every distinct values of the attribute are considered. In order to gather the entropy gain of all these binary tests efficiently, the training data set belonging to the node in consideration is sorted for the values of the continuous attribute and the entropy gains of the binary cut based on each distinct values are calculated in one scan of the sorted data. This process is repeated for each continuous attributes.

II.4.1.4. SLIQ algorithm

SLIQ (Supervised Learning In Quest) developed by IBM's Quest project team, is a decision tree classifier designed to classify large training data [1]. It uses a pre-sorting technique in the tree-growth phase. This helps avoid costly sorting at each node. SLIQ keeps a separate sorted list for each continuous attribute and a separate list called class list.

An entry in the class list corresponds to a data item, and has a class label and name of the node it belongs in the decision tree. An entry in the sorted attribute list has an attribute value and the index of data item in the class list. SLIQ grows the decision tree in breadth-first manner. For each attribute, it scans the corresponding sorted list and calculates entropy values of each distinct value of all the nodes in the frontier of the decision tree simultaneously. After the

entropy values have been calculated for each attribute, one attribute is chosen for a split for each node in the current frontier, and they are expanded to have a new frontier. Then one more scan of the sorted attribute list is performed to update the class list for the new nodes.

While SLIQ handles disk-resident data that are too large to fit in memory, it still requires some information to stay memory-resident which grows in direct proportion to the number of input records, putting a hard-limit on the size of training data. The Quest team has recently designed a new decision-tree-based classification algorithm, called SPRINT (Scalable PaRallelizable INduction of decision Trees) that for the removes all of the memory restrictions.

II.4.1.5. Merits and demerits of Decision Tree methods

Merits

1. Decision trees are able to generate understandable rules.
2. Decision trees perform classification without requiring much computation.
3. Decision trees are able to handle both continuous and categorical variables.
4. Decision trees provide a clear indication of which fields are most important for prediction or classification.

Demerits

Decision trees are less appropriate for estimation tasks where the goal is to predict the values of a continuous variable such as income, blood pressure, or interest rate. Decision trees are also problematic for time-series data unless a lot of effort is put into presenting the data in such a way that trends and sequential patterns are made visible.

1. Error-prone with too many classes.
2. Computationally expensive to train.
3. Trouble with non-rectangular regions.

II.4.1.6. Bayesian Classification

Bayesian classifiers are statistical classifiers. They can predict class membership probabilities, such as the probability that a given tuple belongs to a particular class.

Bayesian classification is based on Bayes' theorem. Studies comparing classification algorithms have found a simple Bayesian classifier known as the naive Bayesian classifier to be comparable in performance with decision tree and selected neural network classifiers. Bayesian classifiers have also exhibited high accuracy and speed when applied to large databases.

Naïve Bayesian classifiers assume that the effect of an attribute value on a given class is independent of the values of the other attributes. This assumption is called class conditional independence. It is made to simplify the computations involved and, in this sense, is considered "naïve." Bayesian belief networks are graphical models, which unlike naïve Bayesian classifiers allow the representation of dependencies among subsets of attributes. Bayesian belief networks can also be used for classification.

So here the core formulation is

$$P(\mathbf{X}|C_i) = \prod_{k=1}^{n} P(x_k|C_i) = P(x_1|C_i) \times P(x_2|C_i) \times \cdots \times P(x_n|C_i).$$

But here, the trick is, since Xi refers to the value of attribute Ak for tuple X. For each attribute, we look at whether the attribute is categorical or continuous-valued. We consider the following:

If Ak is categorical, then P(Xk | Ci) is the number of tuples of class Ci in D having the value Xk for Ak , divided by the number of tuples of class Ci in D.

If Ak is continuous-valued, then we need to do a bit more work, but the calculation is pretty straightforward. A continuous-valued attribute is typically assumed to have a Gaussian distribution with a mean μ and standard deviation σ, defined by

$$g(x, \mu, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}},$$

so that

$$P(x_k|C_i) = g(x_k, \mu_{C_i}, \sigma_{C_i}).$$

If the probability values are zero. We can assume that our training database. D. is so large that adding one to each count that we need would only make a negligible difference in the estimated probability value. yet would conveniently avoid the case of probability values of zero. This technique for probability estimation is known as the Laplacian correction or Laplace estimator.

II.4.2. Prediction

Numeric prediction is the task of predicting continuous (or ordered) values for given input. Some classification techniques (such as back propagation. support vector machines. and k-nearest-neighbor classifiers) can be adapted for prediction. Many problems can be solved by linear regression. and even more can be tackled by applying transformations to the variables so that a nonlinear problem can be converted to a linear one.

Straight-line regression analysis involves a response variable. y. and a single predictor variable. x.

$$y = w_0 + w_1 x.$$

These coefficients can be solved for by the method of least squares. which estimates the best-fitting straight line as the one that minimizes the error between the actual data and the estimate of the line.

$$w_1 = \frac{\sum_{i=1}^{D}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{D}(x_i - \bar{x})^2}$$

$$w_0 = \bar{y} - w_1 \bar{x}$$

When the case becomes to nonlinear, By applying transformations to the variables. we can convert the nonlinear model into a linear one that can then be solved by the method of least squares. For instance. we have

$$y = w_0 + w_1 x + w_2 x^2 + w_3 x^3.$$

To convert this equation to linear form. we define new variables:

$$x_1 = x \qquad x_2 = x^2 \qquad x_3 = x^3$$

Decision tree induction can be adapted so as to predict continuous (ordered) values, rather than class labels. There are two main types of trees for prediction—regression trees and model trees. Each regression tree leaf stores a continuous-valued prediction, which is actually the average value of the predicted attribute for the training tuples that reach the leaf. Since the terms "regression" and "numeric prediction" are used synonymously in statistics, the resulting trees were called "regression trees," even though they did not use any regression equations. By contrast, in model trees, each leaf holds a regression model—a multivariate linear equation for the predicted attribute. Regression and model trees tend to be more accurate than linear regression when the data are not represented well by a simple linear model.

II.4.3. Information Retrieval

Information retrieval is the task, given a set of documents and a user query, of finding the relevant documents. Information retrieval applications require speed, consistency, accuracy and ease of use in retrieving relevant texts to satisfy user queries. These evaluation criteria apply to other text interpretation applications as well.

- ***Accuracy***: The sheer volumes of information now stored in electronic media magnify deficiencies in recall (the percentage of relevant information retrieved. Giving the user reasonable-sized response with high precision can mean missing hundreds of relevant texts.
- ***Speed*** : As the quantity of text that must be searched increases, the speed of searching can become a reserve bottleneck. In practical terms, the need for fast search means that more computational-intensive processing such as NLP techniques must either apply very selectively or run as "batch" indexing tool prior to retrieval.
- ***Consistency*** : Many information retrieval environments require indexing of the text by the groups of indexers or by the authors. This leads to a decrease in accuracy from the inevitable inconsistencies, which automatic processing could help to avoid.
- ***Ease of use*** : The growth of personal computer has made obsolete the traditional model of information retrieval with a trained human intermediary, giving systems responsiveness a high priority.

NetOwl By IsoQuest NetOwl is a family of information retrieval and data extraction software developed by IsoQuest cooperation. This software uses latest and most advanced Natural Language Processing Technology. Some of the members of this software family are as follows.

- Desktop
- Datablade
- Intelligence Server
- Extender
- Discovery Suite
- Extractor

II.4.4. Dimensional Data Model

Dimensional data model is most often used in data warehousing systems. This is different from the 3rd normal form, commonly used for transactional (OLTP) type systems. As you can imagine, the same data would then be stored differently in a dimensional model than in a 3rd normal form model.

To understand dimensional data modeling, let's define some of the terms commonly used in this type of modeling:

Dimension: A category of information. For example, the time dimension.

Attribute: A unique level within a dimension. For example, Month is an attribute in the Time Dimension.

Hierarchy: The specification of levels that represents relationship between different attributes within a dimension. For example, one possible hierarchy in the Time dimension is Year → Quarter → Month → Day.

Fact Table: A fact table is a table that contains the measures of interest. For example, sales amount would be such a measure. This measure is stored in the fact table with the appropriate granularity. For example, it can be sales amount by store by day. In this case, the fact table would contain three columns: A date column, a store column, and a sales amount column.

Lookup Table: The lookup table provides the detailed information about the attributes. For example, the lookup table for the Quarter attribute would include a list of all of the quarters available in the data warehouse. Each row (each quarter) may have several fields, one for the unique ID that identifies the quarter, and one or more additional fields that specifies how that particular quarter is represented on a report (for example, first quarter of 2001 may be represented as "Q1 2001" or "2001 Q1").

A dimensional model includes fact tables and lookup tables. Fact tables connect to one or more lookup tables, but fact tables do not have direct relationships to one another. Dimensions and hierarchies are represented by lookup tables. Attributes are the non-key columns in the lookup tables.

In designing data models for data warehouses / data marts, the most commonly used schema types are Star Schema and Snowflake Schema.

In the ***star schema design***, a single object (the fact table) sits in the middle and is radially connected to other surrounding objects (dimension lookup tables) like a star. Each dimension is represented as a single table. The primary key in each dimension table is related to a forieng key in the fact table.

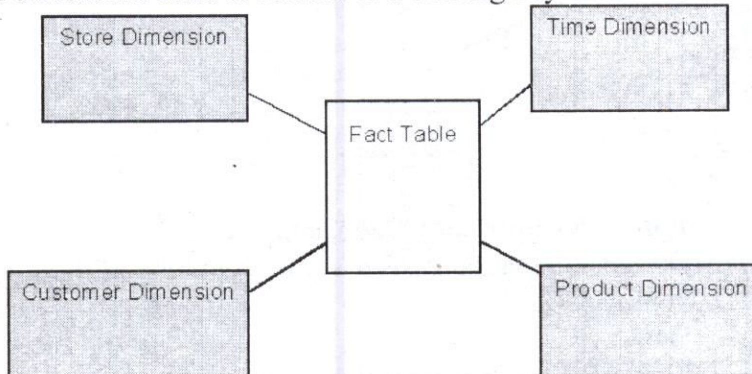


Figure : Sample star schema

All measures in the fact table are related to all the dimensions that fact table is related to. In other words, they all have the same level of granularity.

A star schema can be simple or complex. A simple star consists of one fact table; a complex star can have more than one fact table.

Let's look at an example: Assume our data warehouse keeps store sales data, and the different dimensions are time, store, product, and customer. In this case, the figure on the left represents our star schema. The lines between two tables indicate that there is a primary key / foreign key relationship between the two tables. Note that different dimensions are not related to one another.

The ***snowflake schema*** is an extension of the star schema, where each point of the star explodes into more points. In a star schema, each dimension is represented by a single dimensional table, whereas in a snowflake schema, that dimensional table is normalized into multiple lookup tables, each representing a level in the dimensional hierarchy.

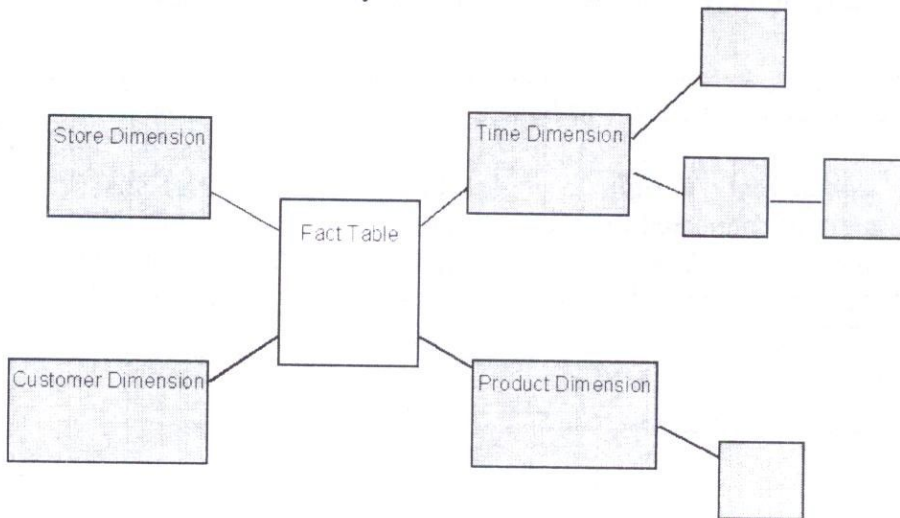


Figure : Sample snowflake schema

For example, the Time Dimension that consists of 2 different hierarchies:

1. Year → Month → Day
2. Week → Day

We will have 4 lookup tables in a snowflake schema: A lookup table for year, a lookup table for month, a lookup table for week, and a lookup table for day. Year is connected to Month, which is then connected to Day. Week is only connected to Day. A sample snowflake schema illustrating the above relationships in the Time Dimension is shown to the right.

The main advantage of the snowflake schema is the improvement in query performance due to minimized disk storage requirements and joining smaller lookup tables. The main disadvantage of the snowflake schemà is the additional maintenance efforts needed due to the increase number of lookup tables.

Building dimensional data model

To build a dimensional database, you start with a dimensional data model. The dimensional data model provides a method for making databases simple and understandable. You can conceive of a dimensional database as a database *cube* of three or four dimensions where users can access a slice of the database along any of its dimensions. To create a dimensional database, you need a model that lets you visualize the data.

Suppose your business sells products in different markets and evaluates the performance over time. It is easy to conceive of this business process as a cube of data, which contains dimensions for time, products, and markets. Figure 35 shows this dimensional model. The various intersections along the lines of the cube would contain the *measures* of the business. The measures correspond to a particular combination of product, market, and time data.

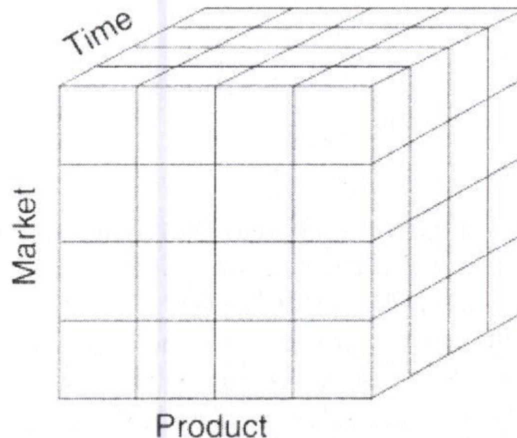


Figure. A Dimensional Model of a Business That Has Time, Product, and Market Dimensions

Another name for the dimensional model is the *star-join schema*. The database designers use this name because the diagram for this model looks like a

star with one central table around which a set of other tables are displayed. The central table is the only table in the schema with multiple joins connecting it to all the other tables. This central table is called the *fact table* and the other tables are called *dimension tables*. The dimension tables all have only a single join that attaches them to the fact table, regardless of the query. Figure shows a simple dimensional model of a business that sells products in different markets and evaluates business performance over time.

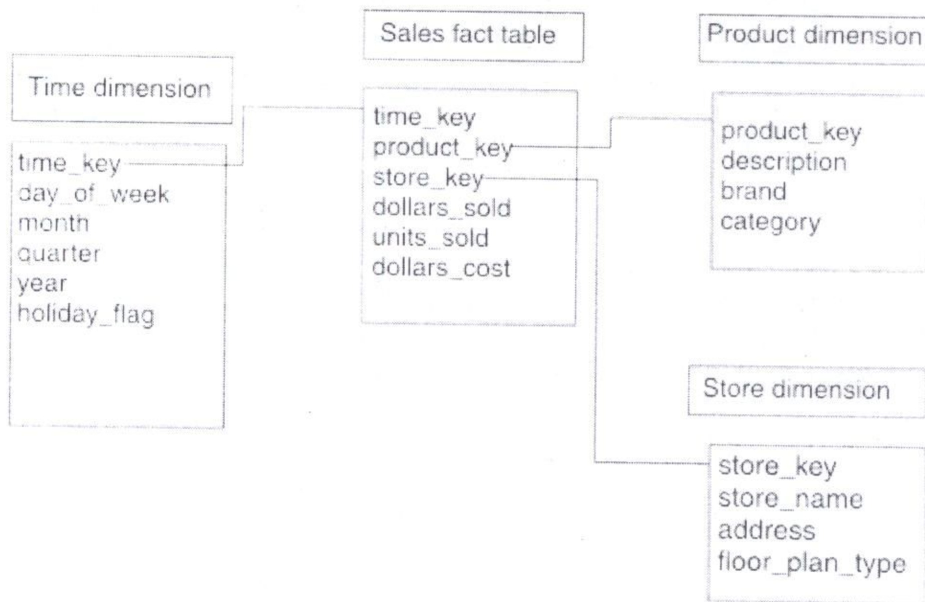


Figure. A Typical Dimensional Model

The fact table stores the measures of the business and points to the key value at the lowest level of each dimension table. The *measures* are quantitative or factual data about the subject. The measures are generally numeric and correspond to the *how much* or *how many* aspects of a question. Examples of measures are price, product sales, product inventory, revenue, and so forth. A measure can be based on a column in a table or it can be calculated.

Table II.4.4.1 shows a fact table whose measures are sums of the units sold, the revenue, and the profit for the sales of that product to that account on that day.

Product Code	Account Code	Day Code	Units Sold	Revenue	Profit
1	5	32104	1	82.12	27.12
3	17	33111	2	171.12	66.00
1	13	32567	1	82.12	27.12

Table II.4.4.1. A Fact Table with Sample Records

Before you design a fact table, you must determine the *granularity* of the fact table. The granularity corresponds to how you define an individual low-level record in that fact table. The granularity might be the individual transaction, a daily snapshot, or a monthly snapshot. The fact table in Table 11 contains one row for every product sold to each account each day. Thus, the granularity of the fact table is expressed as *product by account by day*.

A *dimension* represents a single set of objects or events in the real world. Each dimension that you identify for the data model gets implemented as a dimension table. Dimensions are the qualifiers that make the measures of the fact table meaningful, because they answer the what, when, and where aspects of a question. For example, consider the following business questions, for which the dimensions are italicized:

- What *accounts* produced the highest revenue last *year*?
- What was our profit by *vendor*?
- How many units were sold for each *product*?

In the preceding set of questions, revenue, profit, and units sold are measures (*not* dimensions), as each represents quantitative or factual data.

Dimension Elements

A dimension can define multiple *dimension elements* for different levels of summation. For example, all the elements that relate to the structure of a sales organization might comprise one dimension. Figure shows the dimension elements that the accounts dimension defines.

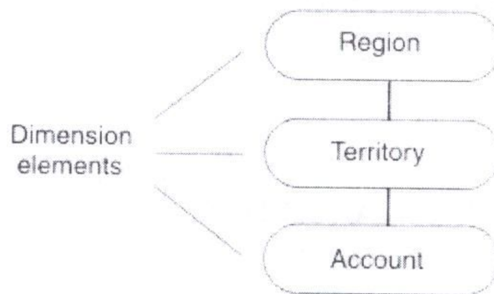


Figure : Dimension Elements in the Accounts Dimension

Dimensions are made up of hierarchies of related elements. Because of the hierarchical aspect of dimensions, users are able to construct queries that access data at a higher level (*roll up*) or lower level (*drill down*) than the previous level of detail. Figure shows the hierarchical relationships of the dimension elements: accounts roll up to territories, and territories roll up to regions. Users can query at different levels of the dimension, depending on the data they want to retrieve. For example, users might perform a query against all regions and then drill down to the territory or account level for detailed information.

Dimension elements are usually stored in the database as numeric codes or short character strings to facilitate joins to other tables. Each dimension element can define multiple dimension attributes, in the same way dimensions can define multiple dimension elements.

Dimension Attributes

A dimension attribute is a column in a dimension table. Each attribute describes a level of summary within a dimension hierarchy. The dimension elements define the hierarchical relationships within a dimension table: the attributes describe dimension elements in terms that are familiar to users. Figure shows the dimension elements and corresponding attributes of the account dimension.

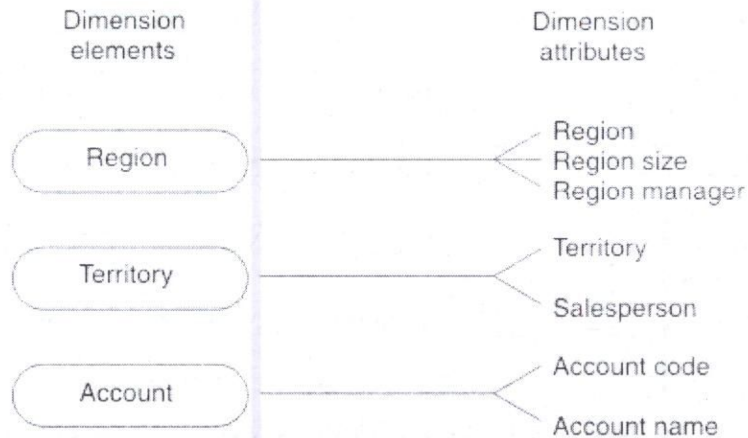


Figure: Attributes That Correspond to the Dimension Elements

Because dimension attributes describe the items in a dimension, they are most useful when they are text.

Tip:

Sometimes during the design process, it is unclear whether a numeric data field from a production data source is a measured fact or an attribute. Generally, if the numeric data field is a measurement that changes each time we sample it, it is a fact. If it is a discretely valued description of something that is more or less constant, it is a dimension attribute.

Dimension Tables

A dimension table is a table that stores the textual descriptions of the dimensions of the business. A dimension table contains an element and an attribute, if appropriate, for each level in the hierarchy. The lowest level of detail that is required for data analysis determines the lowest level in the hierarchy. Levels higher than this base level store redundant data. This de-normalized table reduces the number of joins that are required for a query and makes it easier for users to query at higher levels and then drill down to lower levels of detail. The term *drilling down* means to add row headers from the dimension tables to your query. Table II.4.4.2 shows an example of a dimension table that is based on the account dimension.

Acct Code	Account Name	Territory	Salesman	Region	Region Size	Region Manager
1	Jane's Mfg.	101	B. Adams	Midwest	Over 50	T. Sent
2	TBD Sales	101	B. Adams	Midwest	Over 50	T. Sent
3	Molly's Wares	101	B. Adams	Midwest	Over 50	T. Sent
4	The Golf Co.	201	T. Scott	Midwest	Over 50	T. Sent

Table II.4.4.2.. An Example of a Dimension Table

II.4.5. Pattern matching

In computer science, pattern matching is the act of checking some sequence of tokens for the presence of the constituents of some pattern. In contrast to pattern recognition, the match usually has to be exact. The patterns generally have the form of either sequences or tree structures. Uses of pattern matching include outputting the locations (if any) of a pattern within a token sequence, to output some component of the matched pattern, and to substitute the matching pattern with some other token sequence (I.e., search and replace).

Sequence patterns (e.g., a text string) are often described using regular expressions and matched using techniques such as backtracking.

Tree patterns are used in some programming languages as a general tool to process data based on its structure, e.g., Haskell, ML and the symbolic mathematics language Mathematica have special syntax for expressing tree patterns and a language construct for conditional execution and value retrieval based on it. For simplicity and efficiency reasons, these tree patterns lack some features that are available in regular expressions.

Often it is possible to give alternative patterns that are tried one by one, which yields a powerful conditional programming construct. Pattern matching sometimes include support for guards. Term rewriting and Graph rewriting

languages rely on pattern matching for the fundamental way a program evaluates into a result

In 1977 Robert Boyer and L. Moore published a new exact pattern matching algorithm that had superior logic to existing versions: it performed the character comparison in reverse order to the pattern being searched for and had a method that did not require the full pattern to be searched if a mismatch was found. Legend has it that the original algorithm was coded in PDP-10 assembler.

Computer hardware has changed very considerably since that time yet the fundamental logic is still sound in a different world under very different conditions. Modern personal computers now have the capacity to work on very large sources and often have enough memory to handle those sources without requiring any form of tiling scheme and this capacity very well suits the design of the Boyer Moore exact pattern matching algorithm.

Usage would include tasks like recursively searching files for virus patterns, searching databases for keys or data, text and word processing and any other task that requires handling large amounts of data at very high speed.

The versions presented here are coded in 32 bit Microsoft assembler (MASM) and include a main version that is reasonably close to the original design and two other variant versions that use parts of the original design. The coding is designed for both search speed and mismatch recovery speed in a practical sense that has been determined by direct benchmarking. While all three achieve "sublinearity", the code design is aimed at the delivery of performance, not character count theory based on assumptions related to older ANSI C code.

The code is provided as both source code in three assembler modules and a Microsoft format library that can be used by both MASM and Visual C/C++. To use the modules in VC++, the correct prototypes must be written. The parameters are 32 bit unsigned integers.

How does it work ?

The following example is set up to search for the pattern within the source.

Source "This is a test of the Boyer Moore algorithm."
Pattern "algorithm"

Constructing the Table

A 256 member table is constructed that is initially filled with the length of the pattern which in this case is 9 characters. The 256 members represent the full range of characters in the ASCII character set. A second pass is then made on the table that places a descending count from the original length of the pattern in the ASCII table for each character that occurs.

Algorithm   <- pattern
87654321   <- shift values for each character

The table constructed in this manner allows the algorithm to determine in one access if the character being compared is within the search pattern or not. The first character compared is the end character of the pattern "m" to the corresponding position in the source.

Source "This is a test of the Boyer Moore algorithm."
Pattern "algorithm"

The character being compared is "a" which is within the characters that are in the pattern. Character "a" has a shift of 8 so the pattern is shifted 8 characters right.

Source "This is a test of the Boyer Moore algorithm."
Pattern "algorithm"

II.5 : Estimation of Errors

Evaluation the accuracy of Classifier or Predictor is discussed in this section. The following formulae illustrate how the errors in the classification and predication are estimated.

		Predicted class	
		C_1	C_2
Actual class	C_1	true positives	false negatives
	C_2	false positives	true negatives

$$sensitivity = \frac{t_pos}{pos}$$

$$specificity = \frac{t_neg}{neg}$$

$$precision = \frac{t_pos}{(t_pos + f_pos)}$$

$$accuracy = sensitivity \frac{pos}{(pos + neg)} + specificity \frac{neg}{(pos + neg)}.$$

The true positives, true negatives, false positives, and false negatives are also useful in assessing the costs and benefits (or risks and gains) associated with a classification model. The cost associated with a false negative is far greater than that of a false positive. In such cases, we can outweigh one type of error over another by assigning a different cost to each.

For predictor, the test error (rate), or generalization error, is the average loss over the test set. Thus, we get the following error rates.

Mean absolute error : $\frac{\sum_{i=1}^{d} |y_i - y_i'|}{d}$

Mean squared error : $\frac{\sum_{i=1}^{d} (y_i - y_i')^2}{d}$

Sometimes, we may want the error to be relative to what it would have been if we had just predicted y, the mean value for y from the training data, D. That is, we can normalize the total loss by dividing by the total loss incurred from always predicting the mean. Relative measures of error include:

Relative absolute error : $\dfrac{\sum_{i=1}^{d} |y_i - y_i'|}{\sum_{i=1}^{d} |y_i - \bar{y}|}$

Relative squared error : $\dfrac{\sum_{i=1}^{d} (y_i - y_i')^2}{\sum_{i=1}^{d} (y_i - \bar{y})^2}$

II.6 : Summary

A data mining task can be specified in the form of a data mining query, which is input to the data mining system. A data mining query is defined in terms of data mining task primitives.

Association rule mining searches for interesting relationships among items in a given data set.

A set of items is referred to as an ***itemset***. An ***itemset*** that contains k item is a ***k-itemset***. The set {computer, financial_management_software} is a ***2-itemset***.

The occurrence frequency of an ***itemset*** is the number of transaction that contain the ***itemset***. This is also known, simply, as the frequency, support count or count of the ***itemset***.

Association rule mining is a two-step process:

3. ***Find all frequent itemsets (Frequent itemset generation)***
4. ***Generate strong association rules from the frequent itemsets (Rule Generation)***.

Apriori Algorithm

```
procedure AprioriAlg()
begin
```

```
L1 := {frequent 1-itemsets};
for ( k := 2; Lk-1 0; k++ ) do {
   Ck= apriori-gen(Lk-1) ; // new candidates
for all transactions t in the dataset do {
  for all candidates c Ck contained in t do
          c:count++
  }
  Lk = { c Ck | c:count >= min-support}
}
Answer := k Lk
end
```

Fuzzy sets are sets whose elements have degrees of membership. Fuzzy sets were introduced simultaneously by Lotfi A. Zadeh and Dieter Klaua in 1965 as an extension of the classical notion of set. ***A fuzzy set is a pair (A,m) where A is a set and*** $m : A \to [0, 1]$.

Fuzzy logic is a form of many-valued logic; it deals with reasoning that is approximate rather than fixed and exact. Fuzzy logic corresponds to "degrees of truth", while probabilistic logic corresponds to "probability, likelihood"; as these differ, fuzzy logic and probabilistic logic yield different models of the same real-world situations.

A linguistic variable such as age may have a value such as young or its antonym old. However, the great utility of linguistic variables is that they can be modified via linguistic hedges applied to primary terms. The linguistic hedges can be associated with certain functions.

A fuzzy number is a convex, normalized fuzzy set $\tilde{A} \subseteq \mathbb{R}$ whose membership function is at least segmentally continuous and has the functional value $\mu A(x) = 1$ at precisely one element.

A fuzzy interval is an uncertain set $\tilde{A} \subseteq \mathbb{R}$ with a mean interval whose elements possess the membership function value $\mu A(x) = 1$. As in fuzzy numbers, the membership function must be convex, normalized, atleast segmentally continuous.

The fuzzy relation equation is an equation of the form A · R = B, where A and B are fuzzy sets, R is a fuzzy relation, and A · R stands for the composition of A with R.

Basically classification is a 2-step process; the first step is supervised learning for the sake of the predefined class label for training data set. Second step is classification accuracy evaluation. Likewise data prediction is also 2-step process.

Decision tree is a classifier in the form of a tree structure where each node is either

- A leaf node, indicating a class of instances, or
- A decision node that specifies some test to be carried out on a single attribute value, which one branch and sub tree for each possible outcome of the test.

Dimension: A category of information. For example, the time dimension.

Attribute: A unique level within a dimension. For example, Month is an attribute in the Time Dimension.

Hierarchy: The specification of levels that represents relationship between different attributes within a dimension. For example, one possible hierarchy in the Time dimension is Year → Quarter → Month → Day.

Fact Table: A fact table is a table that contains the measures of interest. For example, sales amount would be such a measure. This measure is stored in the fact table with the appropriate granularity. For example, it can be sales amount by store by day. In this case, the fact table would contain three columns: A date column, a store column, and a sales amount column.

Lookup Table: The lookup table provides the detailed information about the attributes. For example, the lookup table for the Quarter attribute would include a list of all of the quarters available in the data warehouse. Each row (each quarter) may have several fields, one for the unique ID that identifies the quarter, and one or more additional fields that specifies how that particular quarter is

represented on a report (for example, first quarter of 2001 may be represented as "Q1 2001" or "2001 Q1").

A dimensional model includes fact tables and lookup tables. Fact tables connect to one or more lookup tables, but fact tables do not have direct relationships to one another. Dimensions and hierarchies are represented by lookup tables. Attributes are the non-key columns in the lookup tables.

In the ***star schema design***, a single object (the fact table) sits in the middle and is radially connected to other surrounding objects (dimension lookup tables) like a star. Each dimension is represented as a single table. The primary key in each dimension table is related to a forieng key in the fact table.

The ***snowflake schema*** is an extension of the star schema, where each point of the star explodes into more points. In a star schema, each dimension is represented by a single dimensional table, whereas in a snowflake schema, that dimensional table is normalized into multiple lookup tables, each representing a level in the dimensional hierarchy.

A *dimension* represents a single set of objects or events in the real world. Each dimension that you identify for the data model gets implemented as a dimension table.

A dimension can define multiple *dimension elements* for different levels of summation. For example, all the elements that relate to the structure of a sales organization might comprise one dimension.

A dimension attribute is a column in a dimension table. Each attribute describes a level of summary within a dimension hierarchy. The dimension elements define the hierarchical relationships within a dimension table; the attributes describe dimension elements in terms that are familiar to users.

A dimension table is a table that stores the textual descriptions of the dimensions of the business. A dimension table contains an element and an attribute, if appropriate, for each level in the hierarchy.

Evaluation the accuracy of Classifier or Predictor is discussed in this section. The following formulae illustrate how the errors in the classification and predication are estimated.

		Predicted class	
		C_1	C_2
Actual class	C_1	true positives	false negatives
	C_2	false positives	true negatives

$$sensitivity = \frac{t_pos}{pos}$$

$$specificity = \frac{t_neg}{neg}$$

$$precision = \frac{t_pos}{(t_pos + f_pos)}$$

$$accuracy = sensitivity \frac{pos}{(pos + neg)} + specificity \frac{neg}{(pos + neg)}.$$

Mean absolute error : $\dfrac{\sum_{i=1}^{d} |y_i - y_i'|}{d}$

Mean squared error : $\dfrac{\sum_{i=1}^{d} (y_i - y_i')^2}{d}$

Relative absolute error : $\dfrac{\sum_{i=1}^{d} |y_i - y_i'|}{\sum_{i=1}^{d} |y_i - \bar{y}|}$

Relative squared error : $\dfrac{\sum_{i=1}^{d} (y_i - y_i')^2}{\sum_{i=1}^{d} (y_i - \bar{y})^2}$

UNIT III MODELS BASED ON SUMMARIZATION

Structure

III.0 Objectives

In this unit we are going to discuss about the various models based on summarization in data mining. This unit also elaborates in detail about the Baye's Theorem, Chi squared test, Neural Networks and Genetic algorithms. It also discusses the basic concepts of clustering.

At the end of this unit we can

- Understand various models based on summarization
- Got in depth knowledge about Baye's theorem and Chi squared test
- Gain knowledge about clustering
- Elaborate the problems in clustering and the approaches in clustering

III.1 Baye's Theorem

The concept of *conditional probability* is introduced in *Elementary Statistics.* We noted that the conditional probability of an event is a probability obtained with the additional information that some other event has already occurred. We used $P(B|A)$ to denoted the conditional probability of event B occurring, given that event A has already occurred. The following formula was provided for finding $P(B|A)$:

$$P(B|A) = \frac{P(A \text{ and } B)}{P(A)}$$

In addition to the above formal rule, the textbook also included this "intuitive approach for finding a conditional probability":

> The conditional probability of B given A can be found by assuming that event A has occurred and, working under that assumption, calculating the probability that event B will occur.

In this section we extend the discussion of conditional probability to include applications of *Bayes' theorem* (or *Bayes' rule*), which we use for revising a probability value based on additional information that is later obtained. One key to understanding the essence of Bayes' theorem is to recognize that we are dealing with *sequential* events, whereby new additional information is obtained for a subsequent event, and that new information is used to revise the probability of the initial event. In this context, the terms *prior probability* and *posterior probability* are commonly used.

Definitions

A **prior probability** is an initial probability value originally obtained before any additional information is obtained.

A **posterior probability** is a probability value that has been revised by using additional information that is later obtained.

Example 1

The Gallup organization randomly selects an adult American for a survey about credit card usage. Use subjective probabilities to estimate the following.

a. What is the probability that the selected subject is a male?
b. After selecting a subject, it is later learned that this person was smoking a cigar during the interview. What is the probability that the selected subject is a male?

c. Which of the preceding two results is a prior probability? Which is a posterior probability?

Solution

a. Roughly half of all Americans are males, so we estimate the probability of selecting a male subject to be 0.5. Denoting a male by M, we can express this probability as follows: $P(\mathrm{M}) = 0.5$.

b. Although some women smoke cigars, the vast majority of cigar smokers are males. A reasonable guess is that 85% of cigar smokers are males. Based on this additional subsequent information that the survey respondent was smoking a cigar, we estimate the probability of this person being a male as 0.85. Denoting a male by M and denoting a cigar smoker by C, we can express this result as follows: $P(\mathrm{M} \mid \mathrm{C}) = 0.85$.

c. In part (a), the value of 0.5 is the initial probability, so we refer to it as the prior probability. Because the probability of 0.85 in part (b) is a revised probability based on the additional information that the survey subject was smoking a cigar, this value of 0.85 is referred to a posterior probability.

The Reverend Thomas Bayes [1701 (approximately) – 1761] was an English minister and mathematician. Although none of his work was published during his lifetime, later (posterior?) publications included the following theorem (or rule) that he developed for determining probabilities of events by incorporating information about subsequent events.

Bayes' Theorem

The probability of event A, given that event B has subsequently occurred, is

$$P(A \mid B) = \frac{P(A) \cdot P(B \mid A)}{[P(A) \cdot P(B \mid A)] + [P(\overline{A}) \cdot P(B \mid \overline{A})]}$$

That's a formidable expression, but we will simplify its calculation. See the following example, which illustrates use of the above expression, but also see the alternative method based on a more intuitive application of Bayes' theorem.

Example 2

In Orange County, 51% of the adults are males. (It doesn't take too much advanced mathematics to deduce that the other 49% are females.) One adult is randomly selected for a survey involving credit card usage.

a. Find the prior probability that the selected person is a male.

b. It is later learned that the selected survey subject was smoking a cigar. Also, 9.5% of males smoke cigars, whereas 1.7% of females smoke cigars (based on data from the Substance Abuse and Mental Health Services Administration).

Use this additional information to find the probability that the selected subject is a male.

Solution

Let's use the following notation:

M = male	$\overline{M}$ = female (or not male)
C = cigar smoker	$\overline{C}$ = not a cigar smoker.

a. Before using the information given in part b, we know only that 51% of the adults in Orange County are males, so the probability of randomly selecting an adult and getting a male is given by $P(M) = 0.51$.

b. Based on the additional given information, we have the following:

$P(M) = 0.51$ because 51% of the adults are males

$P(\overline{M}) = 0.49$ because 49% of the adults are females (not males)

$P(C|M) = 0.095$ because 9.5% of the males smoke cigars (That is, the probability of getting someone who smokes cigars, given that the person is a male, is 0.095.)

$P(C|\overline{M}) = 0.017$. because 1.7% of the females smoke cigars (That is, the probability of getting someone who smokes cigars, given that the person is a female, is 0.017.)

Let's now apply Bayes' theorem by using the preceding formula with M in place of A, and C in place of B. We get the following result:

$$P(M|C) = \frac{P(M)\cdot P(C|M)}{[P(M)\cdot P(C|M)]+[P(\overline{M})\cdot P(C|\overline{M})]}$$

$$= \frac{0.51\cdot 0.095}{[0.51\cdot 0.095]+[0.49\cdot 0.017]}$$

$$= 0.85329341$$

$$= 0.853 \text{ (rounded)}$$

Before we knew that the survey subject smoked a cigar, there is a 0.51 probability that the survey subject is male (because 51% of the adults in Orange County are males). However, after learning that the subject smoked a cigar, we revised the probability to 0.853. There is a 0.853 probability that the cigar−smoking respondent is a male. This makes sense, because the likelihood of a male increases dramatically with the additional information that the subject smokes cigars (because so many more males smoke cigars than females).

III.2 Chi-square statistics regression

The *chi-square* (chi, the Greek letter pronounced "kye") statistic is a nonparametric statistical technique used to determine if a distribution of observed frequencies differs from the theoretical expected frequencies. Chi-square statistics use nominal (categorical) or ordinal level data, thus instead of using means and variances, this test uses frequencies.

The value of the chi-square statistic is given by

$$\chi^2 = \sum \frac{(O-E)^2}{E}$$

where χ^2 is the chi-square statistic, O is the observed frequency and E is the expected frequency.

Generally the *chi-squared statistic* summarizes the discrepancies between the expected number of times each outcome occurs (assuming that the model is true) and the observed number of times each outcome occurs, by summing the squares of the discrepancies, normalized by the expected numbers, over all the categories. Data used in a chi-square analysis has to satisfy the following conditions:

1. Randomly drawn from the population,
2. reported in raw counts of frequency,
3. measured variables must be independent,
4. observed frequencies cannot be too small, and
5. Values of independent and dependent variables must be mutually exclusive.

There are two types of chi-square test.

- *The Chi-square test for goodness of fit* which compares the expected and observed values to determine how well an experimenter's predictions fit the data.
- *The Chi-square test for independence* which compares two sets of categories to determine whether the two groups are distributed differently among the categories.

III.2.1. Chi-square test for Goodness of Fit

Goodness of fit means how well a statistical model fits a set of observations. A measure of *goodness of fit* typically summarizes the discrepancy between observed values and the values expected under the model in question. Such measures can be used in statistical hypothesis testing, e.g., to test for normality of residuals, to test whether two samples are drawn from identical distributions.

Example

Suppose a coin is tossed 100 times, the outcomes would be expected to be 50 heads and 50 tails. If 47 heads and 53 tails are observed instead, does this deviation occur because the coin is biased, or is it by chance?

Establish Hypotheses

The Null hypothesis for the above experiment is that the observed values are close to the predicted values. The alternative hypothesis is that they are not close to the predicted values. These hypotheses hold for all Chi-square *goodness of fit* tests. Thus in this case the null and alternative hypotheses correspond to:

Null hypothesis: The coin is fair

Alternative hypothesis: The coin is biased

	Heads	Tails
Observed	47	53
Expected	50	50

Table 1: Table 1 Tabulated results of Observed and Expected frequencies

Calculate the chi-square statistic

We calculate chi-square by substituting values for *O* and *E*

For Heads: *X2* = (47-50)2/50 = 0.18

For Tails *X2* = (53-50)2/50 = 0.18

The sum of these categories is 0.18 + 0.18 = 0.36

Assessing significance levels

Significance of the *chi-square test for goodness of fit* value is established by calculating the *degree of freedom* v (the Greek letter nu) and by using the chi-

square distribution table. The v in a *chi-square goodness of fit test* is equal to the number of categories, c, minus one (v= c-1). This is done in order to check if the null hypothesis is valid or not, by looking at the critical chi-square value from the table that corresponds to the calculated v.

If the calculated Chi-square is greater than the value in the table, then the null hypothesis is rejected and it is concluded that the predictions made were incorrect. In the above experiment, v= (2-1) = 1. The critical value for a chi-square for this example at a = 0.05 and v =1 is 3.84 which is greater than χ^2 =0.36. Therefore the null hypothesis is not rejected; hence the coin toss was fair.

III.2.2. Chi-square test for Independence

The chi-square test for independence is used to determine the relationship between two variables of a sample. In this context independence means that the two factors are not related. Typically in social science research, we're interested in finding factors which are related, e.g. education and income, occupation and prestige, age and voting behavior.

Example:

We want to know whether boys or girls get into trouble more often in school. Below is the table documenting the frequency of boys and girls who got into trouble in school?

	Got into trouble *(Observed)*	Not in trouble *(Observed)*	Total	Got· into trouble *(Expected)*	Not in trouble *(Expected)*
Boys	46	71	117	*(40.97)*	*(76.02)*
Girls	37	83	120	*(42.03)*	*(77.97)*
Total	83	154	237		

Table 2: Tabulated results of the Observed and Expected frequency

To examine statistically whether boys got in trouble more often in school, we need to establish hypotheses for the question.

Establish Hypotheses

The *null hypothesis* is that the two variables are independent or in this particular case is that the likelihood of getting in trouble is the same for boys and

girls. The alternative hypothesis to be tested is that the likelihood of getting in trouble is not the same for boys and girls.

Cautionary Note

It is important to keep in mind that the chi-square test for independence only tests whether two variables are independent or not, it cannot address questions of which is greater or less. Using the chi-square test for independence, it cannot be evaluated directly from the hypothesis that gets more in trouble between boys and girls.

Calculate the expected value for each cell of the table

As with the *goodness of fit* example described earlier, the key idea of the chi-square test for independence is a comparison of observed and expected values. In the case of tabular data, however, we usually do not know what the distribution should look like (as we did with tossing the coin). Rather expected values are calculated based on the row and column totals from the table.

The expected value for each cell of the table can be calculated using the following equation:

expected value = Row total * Column total / Total for table (2)

The expected values (in parentheses, italics and bold) for each cell are also presented in Table 2.

Calculate Chi-square statistic

With the values in Table 2, the chi-square statistic can be calculated using Equation 1 as follows:

$$\chi^2 = (46-40.97)^2 / 40.97 + (37-42.03)^2 / 42.03 + (71-76.03)^2 / 76.03 + (83-77.97)^2 / 77.97 = 1.87$$

Assessing significance levels

In the *chi-square test for independence* the degree of freedom is equal to the number of columns in the table minus one multiplied by the number of rows in the table minus one.

i.e. dof = (r-1)(c-1) = 1.

Thus the value calculated from the formula above is compared with values in the chi-square distribution table. The value returned from the table is p< 20%. Therefore the null hypothesis is not rejected; hence boys are not significantly more likely to get in trouble in school than girls.

III.3 Decision Trees

Decision trees are powerful and popular tool for classification and prediction. Decision trees represent rules. Decision tree is a classifier in the form of a tree structure where each node is either

- A leaf node, indicating a class of instances, or
- A decision node that specifies some test to be carried our on a single attribute value, which one branch and sub tree for each possible outcome of the test.

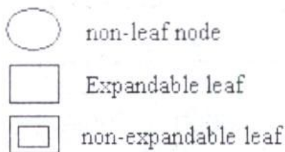
A decision tree can be used to classify and instance by starting at the root of the tree and moving through it until a leaf node, which provides the classification of the instance.

1) Constructing Decision Trees

Decision tree programs construct a decision tree T from a set of training cases. The original idea of construction of decision trees goes back to the work of Hoveland and Hunt on Concept Learning Systems (CLS) in the late 1950s.

The algorithm consists of five steps.

1. T ← the whole training set. Create a T node.
2. If all the samples in T are positive, create a "P" node with T as its parent and stop.
3. If all the samples in t are negative, create an "N" node with T as its parent and stop.
4. Select an attribute X with values v1, v2... vN and partition T into subsets T1, T2... TN according their values on X. Create N nodes Ti (i=1, 2,...., N) with T as their parent and X=vi as the label of the branch from T to Ti.
5. For each Ti do: T ← Ti and go to step 2.



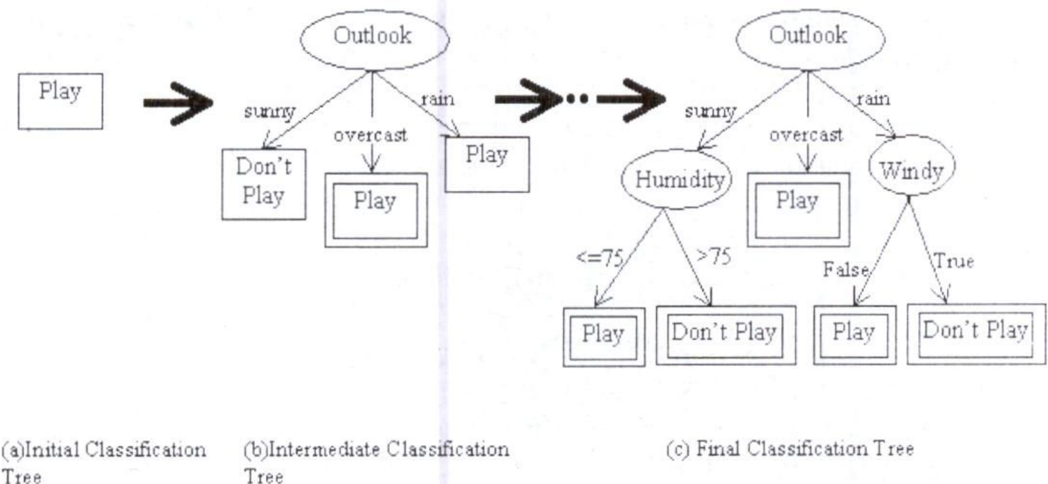


Figure : Demonstration of Hunt's method : An example for a simple decision tree

Decision tree induction is a typical inductive approach to learn knowledge on classification. The key requirements to do mining with decision tree are:

- → Predefined classes: The categories to which cases are to be assigned must have been established beforehand (supervised data).
- → Discrete classes: A case does or does not belong to a particular class. and there must be for more cases than classes.
- → Sufficient data: Usually hundreds or even thousands of training cases.
- → "Logical" classification model: Classifier that can be only expressed as decision tree or set of production rules.

III.4 Neural Networks

Neural Networks are analytic techniques modeled after the (hypothesized) processes of learning in the cognitive system and the neurological functions of the brain and capable of predicting new observations (on specific variables) from other observations (on the same or other variables) after executing a process of so-called learning from existing data. Neural Networks is one of the Data Mining techniques.

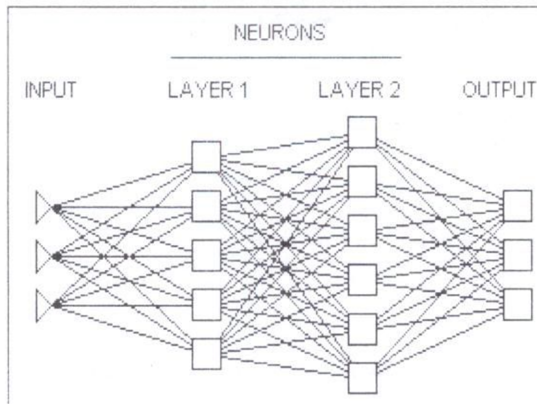


Figure : Neural Networks

The first step is to design a specific network architecture (that includes a specific number of "layers" each consisting of a certain number of "neurons"). The size and structure of the network needs to match the nature (e.g., the formal complexity) of the investigated phenomenon. Because the latter is obviously not known very well at this early stage, this task is not easy and often involves multiple "trials and errors." (Now, there is, however, neural network software that applies artificial intelligence techniques to aid in that tedious task and finds "the best" network architecture.)

The new network is then subjected to the process of "training." In that phase, neurons apply an iterative process to the number of inputs (variables) to adjust the weights of the network in order to optimally predict (in traditional terms one could say, find a "fit" to) the sample data on which the "training" is performed. After the phase of learning from an existing data set, the new network is ready and it can then be used to generate predictions.

The resulting "network" developed in the process of "learning" represents a pattern detected in the data. Thus, in this approach, the "network" is the functional equivalent of a model of relations between variables in the traditional model building approach. However, unlike in the traditional models, in the "network," those relations cannot be articulated in the usual terms used in statistics or methodology to describe relations between variables (such as, for example, "A is positively correlated with B but only for observations where the value of C is low and D is high"). Some neural networks can produce highly accurate predictions; they represent, however, a typical a-theoretical (one can say, "a black box") research approach. That approach is concerned only with practical considerations, that is, with the predictive validity of the solution and its applied relevance and not with the nature of the underlying mechanism or its relevance for any "theory" of the underlying phenomena.

However, it should be mentioned that Neural Network techniques can also be used as a component of analyses designed to build explanatory models because Neural Networks can help explore data sets in search for relevant variables or groups of variables; the results of such explorations can then facilitate the process of model building. Moreover, now there is neural network software that uses sophisticated algorithms to search for the most relevant input variables, thus potentially contributing directly to the model building process.

One of the major advantages of neural networks is that, theoretically, they are capable of approximating any continuous function, and thus the researcher does not need to have any hypotheses about the underlying model, or even to some extent, which variables matter. An important disadvantage, however, is that the final solution depends on the initial conditions of the network, and, as stated before, it is virtually impossible to "interpret" the solution in traditional, analytic terms, such as those used to build theories that explain phenomena.

Some authors stress the fact that neural networks use, or one should say, are expected to use, massively parallel computation models. For example Haykin (1994) defines neural network as:

"A massively parallel distributed processor that has a natural propensity for storing experiential knowledge and making it available for use. It resembles

the brain in two respects: (1) Knowledge is acquired by the network through a learning process, and (2) Interneuron connection strengths known as synaptic weights are used to store the knowledge."

However, as Ripley (1996) points out, the vast majority of contemporary neural network applications run on single-processor computers and he argues that a large speed-up can be achieved not only by developing software that will take advantage of multiprocessor hardware by also by designing better (more efficient) learning algorithms.

Merits

Neural networks is that, theoretically, they are capable of approximating any continuous function, and thus the researcher does not need to have any hypotheses about the underlying model, or even to some extent, which variables matter.

Demerits

The final solution depends on the initial conditions of the network. It is virtually impossible to "interpret" the solution in traditional, analytic terms, such as those used to build theories that explain phenomena.

III.5 Genetic Algorithm

Genetic algorithms attempt to incorporate ideas of natural evolution. In general, genetic learning starts as follows. An initial population is created consisting of randomly generated rules. Each rule can be represented by a string of bits.

As a simple example, supposed that samples in a given training set are described by two Boolean attributes, A1 and A2, and that there are two classes, C1 and C2. The rule "IF A1 AND NOT A2 THEN C2" can be encoded as the bit string "100", where the two leftmost bits represent attributes A1 and A2 respectively, and the rightmost bit represents the class. Similarly, the rule "IF NOT A1 AND NOT A2 THEN C1", can be encoded as "001". If an attribute has k values, where k>2, then k bits may be used to encode the attribute's values. Classes can be encoded in a similar fashion.

Genetic algorithms / Evolutionary algorithms are based on Darwin's theory of survival of the fittest; a new population is formed to consist of the fittest rules in the current population, as well as offspring of these rules. Typically, the fitness of a rule is assessed by its classification accuracy on a set of training samples.

Offspring are created by applying genetic operators such as crossover and mutation. In crossover, substrings from pairs of rules are swapped to form new pairs of rules. In mutation, randomly selected bits in a rule's string are inverted.

The process of generating new populations based on prior populations of rules continues until a population, P, evolves where each rule in P satisfies a pre specified fitness threshold. Genetic algorithms are easily parallelizable and have been used for classification as well as other optimization problems. In data mining, they may be used to evaluate the fitness of other algorithms.

III.6 Cluster Analysis

The term cluster analysis (first used by Tryon, 1939) encompasses a number of different algorithms and methods for grouping objects of similar kind into respective categories. A general question facing researchers in many areas of inquiry is how to organize observed data into meaningful structures, that is, to develop taxonomies. In other words cluster analysis is an exploratory data analysis tool which aims at sorting different objects into groups in a way that the degree of association between two objects is maximal if they belong to the same group and minimal otherwise.

Given the above, cluster analysis can be used to discover structures in data without providing an explanation/interpretation. In other words, cluster analysis simply discovers structures in data without explaining why they exist.

We deal with clustering in almost every aspect of daily life. For example, a group of diners sharing the same table in a restaurant may be regarded as a cluster of people. In food stores items of similar nature, such as different types of meat or vegetables are displayed in the same or nearby locations. There is a

countless number of examples in which clustering plays an important role. For instance, biologists have to organize the different species of animals before a meaningful description of the differences between animals is possible. According to the modern system employed in biology, man belongs to the primates, the mammals, the amniotes, the vertebrates, and the animals. Note how in this classification, the higher the level of aggregation the less similar are the members in the respective class.

Man has more in common with all other primates (e.g., apes) than it does with the more "distant" members of the mammals (e.g., dogs), etc. For a review of the general categories of cluster analysis methods, see Joining (Tree Clustering), Two-way Joining (Block Clustering), and k-Means Clustering. In short, whatever the nature of your business is, sooner or later you will run into a clustering problem of one form or another.

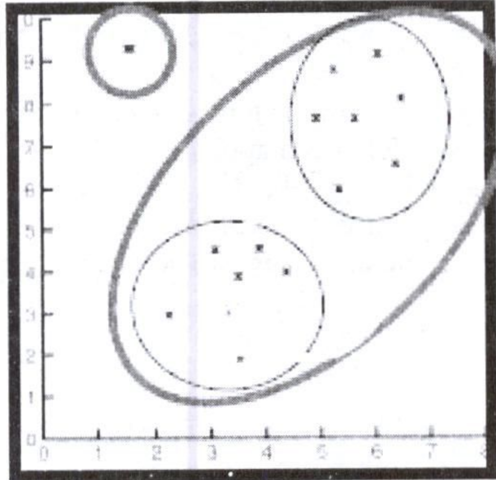
Cluster Outlier

An outlying observation, or outlier, is one that appears to deviate markedly from other members of the sample in which it occurs.

Outliers can occur by chance in any distribution, but they are often indicative either of measurement error or that the population has a heavy-tailed distribution. In the former case one wishes to discard them or use statistics that are robust to outliers, while in the latter case they indicate that the distribution has high kurtosis and that one should be very cautious in using tools or intuitions that assume a normal distribution. A frequent cause of outliers is a mixture of two distributions, which may be two distinct sub-populations, or may indicate 'correct trial' versus 'measurement error'; this is modeled by a mixture model.

In larger samplings of data, some data points will be further away from the sample mean than what is deemed reasonable. This can be due to incidental systematic error or flaws in the theory that generated an assumed family of probability distributions, or it may be that some observations are far from the center of the data. Outlier points can therefore indicate faulty data, erroneous procedures, or areas where a certain theory might not be valid. However, in large samples, a small number of outliers is to be expected (and not due to any anomalous condition).

Outliers, being the most extreme observations, may include the sample maximum or sample minimum, or both, depending on whether they are extremely high or low. However, the sample maximum and minimum are not always outliers because they may not be unusually far from other observations.



What are the "best" two Cluster?

Figure : Impact of Outliers on Clustering

Clustering Vs Classification

- Clustering
  - Unsupervised
  - Input
    - Clustering algorithm
    - Similarity measure
    - Number of clusters
  - No specific information for each document
- Classification
  - Supervised
  - Each document is labeled with a class
  - Build a classifier that assigns documents to one of the classes

Clustering issues

The issues of clustering are listed below

→ Outlier handling
→ Dynamic data
→ Interpreting results
→ Evaluating results
→ Number of clusters
→ Data to be used
→ Scalability

Problems in Clustering

- Given a database D={t1,t2,….,tn} of tuples and an integer value k, the ***Clustering Problem*** is to define a mapping f:D□{1,..,k} where each ti is assigned to one cluster Kj, 1<=j<=k.
- A ***Cluster***, Kj, contains precisely those tuples mapped to it.
- Unlike classification problem, clusters are not known a priori

Cluster Parameters

$$centroid = C_m = \frac{\sum_{i=1}^{N}(t_{mi})}{N}$$

$$radius = R_m = \sqrt{\frac{\sum_{i=1}^{N}(t_{mi} - C_m)^2}{N}}$$

$$diameter = D_m = \sqrt{\frac{\sum_{i=1}^{N}\sum_{j=1}^{N}(t_{mi} - t_{mj})^2}{(N)(N-1)}}$$

Types of Clustering

- ***Hierarchical*** – Creates Tree of clusterings
  – Agglomerative (bottom up)
  – Divisive (top down)
- ***Partitional*** – One set of clusters created.

III.7 Summary

A **prior probability** is an initial probability value originally obtained before any additional information is obtained.

A **posterior probability** is a probability value that has been revised by using additional information that is later obtained.

Bayes' Theorem

The probability of event *A*, given that event *B* has subsequently occurred, is

$$P(A|B) = \frac{P(A)\cdot P(B|A)}{[P(A)\cdot P(B|A)] + [P(\overline{A})\cdot P(B|\overline{A})]}$$

The *chi-square* (chi, the Greek letter pronounced "kye") statistic is a nonparametric statistical technique used to determine if a distribution of observed frequencies differs from the theoretical expected frequencies. Chi-square statistics use nominal (categorical) or ordinal level data, thus instead of using means and variances, this test uses frequencies.

The value of the chi-square statistic is given by

$$\chi^2 = \sum \frac{(O-E)^2}{E}$$

where χ^2 is the chi-square statistic, O is the observed frequency and E is the expected frequency.

Data used in a chi-square analysis has to satisfy the following conditions:

1. Randomly drawn from the population,
2. reported in raw counts of frequency,
3. measured variables must be independent,
4. observed frequencies cannot be too small, and
5. Values of independent and dependent variables must be mutually exclusive.

There are two types of chi-square test.

- *The Chi-square test for goodness of fit* which compares the expected and observed values to determine how well an experimenter's predictions fit the data.
- *The Chi-square test for independence* which compares two sets of categories to determine whether the two groups are distributed differently among the categories.

Goodness of fit means how well a statistical model fits a set of observations. A measure of *goodness of fit* typically summarizes the discrepancy between observed values and the values expected under the model in question. Such measures can be used in statistical hypothesis testing, e.g., to test for normality of residuals, to test whether two samples are drawn from identical distributions. The chi-square test for independence is used to determine the relationship between two variables of a sample.

The key requirements to do mining with decision tree are:

→ Predefined classes: The categories to which cases are to be assigned must have been established beforehand (supervised data).

→ Discrete classes: A case does or does not belong to a particular class, and there must be for more cases than classes.

→ Sufficient data: Usually hundreds or even thousands of training cases.

→ "Logical" classification model: Classifier that can be only expressed as decision tree or set of production rules.

Neural Networks are analytic techniques modeled after the (hypothesized) processes of learning in the cognitive system and the neurological functions of the brain and capable of predicting new observations (on specific variables) from other observations (on the same or other variables) after executing a process of so-called learning from existing data.

Genetic algorithms / Evolutionary algorithms are based on Darwin's theory of survival of the fitte : a new population is formed to consist of the fittest rules in the current populat n, as well as offspring of these rules. Typically, the fitness of a rule is assesse by its classification accuracy on a set of training samples.

Clustering Vs Cla ification

- Clustering
  - Unsu ervised
  - Input
    - Clustering algorithm
    - Similarity measure
    - Number of clusters
  - No specific information for each document

- Classification
    - Supervised
    - Each document is labeled with a class
    - Build a classifier that assigns documents to one of the classes

Clustering issues

The issues of clustering are listed below

→ Outlier handling
→ Dynamic data
→ Interpreting results
→ Evaluating results
→ Number of clusters
→ Data to be used
→ Scalability

Problems in Clustering

- Given a database D={t1,t2,….,tn} of tuples and an integer value k, the ***Clustering Problem*** is to define a mapping f:D ⃞ {1,...,k} where each ti is assigned to one cluster Kj, 1<=j<=k.
- A ***Cluster***, Kj, contains precisely those tuples mapped to it.
- Unlike classification problem, clusters are not known a priori

Cluster Parameters

$$centroid = C_m = \frac{\sum_{i=1}^{N}(t_{mi})}{N}$$

$$radius = R_m = \sqrt{\frac{\sum_{i=1}^{N}(t_{mi} - C_m)^2}{N}}$$

$$diameter = D_m = \sqrt{\frac{\sum_{i=1}^{N}\sum_{j=1}^{N}(t_{mi} - t_{mj})^2}{(N)(N-1)}}$$

Types of Clustering

- ***Hierarchical*** – Creates Tree of clusterings
    – Agglomerative (bottom up)
    – Divisive (top down)
- ***Partitional*** – One set of clusters created.

UNIT IV CLUSTERING ALGORITHMS

Structure

IV.0 Objectives

In this unit we are going to discuss about the basic concepts related to clustering algorithms. This unit also elaborates how data mining acts as the core process of knowledge discovery from databases. It also discusses about hierarchical and partitonal algorithms in clustering.

At the end of this unit we can

→ Understand the concept of clustering
→ Identify the kinds of clustering
→ Gain knowledge about the K-means nearest neighbor

IV.1 Introduction to Clustering Algorithms

Cluster analysis classifies a set of observations into two or more mutually exclusive unknown groups based on combinations of interval variables. The purpose of cluster analysis is to discover a system of organizing observations. usually people, into groups. Where members of the groups share properties in common.

It is cognitively easier for people to predict behavior or properties of people or objects based on group membership, all of whom share similar properties. It is generally cognitively difficult to deal with individuals and

predict behavior or properties based on observations of other behaviors or properties.

For example, a person might wish to predict how an animal would respond to an invitation to go for a walk. He or she could be given information about the size and weight of the animal, top speed, average number of hours spent sleeping per day and so forth and then combines that information into a prediction of behavior. Alternatively, the person could be told that an animal is either a cat or a dog. The latter information allows a much broader range of behaviors to be predicted. The trick in cluster analysis is to collect information and combine it in ways that allow classification into useful groups, such as dog or cat.

Cluster analysis classifies unknown groups while discriminant function analysis classifies known groups. The procedure for doing a discriminant function analysis is well established. There are few options, other than type of output, that need to be specified when doing a discriminant function analysis. Cluster analysis, on the other hand, allows many choices about the nature of the algorithm for combining groups. Each choice may result in a different grouping structure.

IV.2 Hierarchical and Partitional Algorithm

Hierarchical clustering creates a hierarchy of clusters which may be represented in a tree structure called a dendrogram. The root of the tree consists of a single cluster containing all observations, and the leaves correspond to individual observations.

Algorithms for hierarchical clustering are generally either agglomerative, in which one starts at the leaves and successively merges clusters together; or divisive, in which one starts at the root and recursively splits the clusters.

Any non-negative-valued function may be used as a measure of similarity between pairs of observations. The choice of which clusters to merge or split is

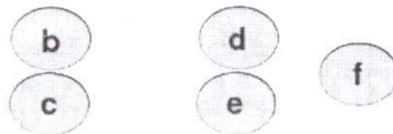
determined by a linkage criterion, which is a function of the pair-wise distances between observations.

Cutting the tree at a given height will give a clustering at a selected precision. In the following example, cutting after the second row will yield clusters {a} {b c} {d e} {f}. Cutting after the third row will yield clusters {a} {b c} {d e f}, which is a coarser clustering, with a smaller number of larger clusters.

Agglomerative hierarchical clustering

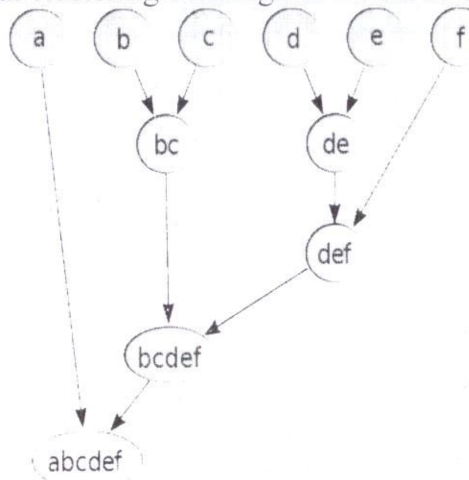
For example, suppose this data is to be clustered, and the Euclidean distance is the distance metric.





Raw data

The hierarchical clustering dendrogram would be as such:



Traditional representation

This method builds the hierarchy from the individual elements by progressively merging clusters. In our example, we have six elements {a} {b} {c} {d} {e} and {f}. The first step is to determine which elements to merge in a cluster. Usually, we want to take the two closest elements, according to the chosen distance.

Optionally, one can also construct a distance matrix at this stage, where the number in the i-th row j-th column is the distance between the i-th and j-th elements. Then, as clustering progresses, rows and columns are merged as the clusters are merged and the distances updated. This is a common way to implement this type of clustering, and has the benefit of caching distances between clusters. A simple agglomerative clustering algorithm is described in the single-linkage clustering page; it can easily be adapted to different types of linkage (see below).

Suppose we have merged the two closest elements b and c, we now have the following clusters {a}, {b, c}, {d}, {e} and {f}, and want to merge them further. To do that, we need to take the distance between {a} and {b c}, and therefore define the distance between two clusters. Usually the distance between two clusters $\mathcal{A}$and $\mathcal{B}$is one of the following:

- The maximum distance between elements of each cluster (also called complete-linkage clustering):
  $\max\{\, d(x,y) : x \in \mathcal{A},\, y \in \mathcal{B}\,\}.$
- The minimum distance between elements of each cluster (also called single-linkage clustering):
  $\min\{\, d(x,y) : x \in \mathcal{A},\, y \in \mathcal{B}\,\}.$
- The mean distance between elements of each cluster (also called average linkage clustering, used e.g. in UPGMA):
  $$\frac{1}{|\mathcal{A}|\cdot|\mathcal{B}|}\sum_{x\in\mathcal{A}}\sum_{y\in\mathcal{B}} d(x,y).$$
- The sum of all intra-cluster variance.
- The increase in variance for the cluster being merged (Ward's method: introduced in J. H. Ward, Jr. 1963).[1]

- The probability that candidate clusters spawn from the same distribution function (V-linkage).

Each agglomeration occurs at a greater distance between clusters than the previous agglomeration, and one can decide to stop clustering either when the clusters are too far apart to be merged (distance criterion) or when there is a sufficiently small number of clusters (number criterion).

Concept clustering

Another variation of the agglomerative clustering approach is conceptual clustering.

The k-means algorithm assigns each point to the cluster whose center (also called centroid) is nearest. The center is the average of all the points in the cluster — that is, its coordinates are the arithmetic mean for each dimension separately over all the points in the cluster.

Example: The data set has three dimensions and the cluster has two points: X = (x1,x2,x3) and Y = (y1,y2,y3). Then the centroid Z becomes Z = (z1,z2,z3), where $z_1 = \frac{x_1 + y_1}{2}$, $z_2 = \frac{x_2 + y_2}{2}$ and $z_3 = \frac{x_3 + y_3}{2}$

The algorithm steps are:

- Choose the number of clusters, k.
- Randomly generate k clusters and determine the cluster centers, or directly generate k random points as cluster centers.
- Assign each point to the nearest cluster center, where "nearest" is defined with respect to one of the distance measures discussed above.
- Recompute the new cluster centers.
- Repeat the two previous steps until some convergence criterion is met (usually that the assignment hasn't changed).

The main advantages of this algorithm are its simplicity and speed which allows it to run on large datasets. Its disadvantage is that it does not yield the same result with each run, since the resulting clusters depend on the initial random assignments (the k-means++ algorithm addresses this problem by seeking to choose better starting clusters). It minimizes intra-cluster variance.

but does not ensure that the result has a global minimum of variance. Another disadvantage is the requirement for the concept of a mean to be definable which is not always the case. For such datasets the k-medoids variants is appropriate. An alternative, using a different criterion for which points are best assigned to which centre is k-medians clustering.

Fuzzy clustering

In fuzzy clustering, each point has a degree of belonging to clusters, as in fuzzy logic, rather than belonging completely to just one cluster. Thus, points on the edge of a cluster, may be in the cluster to a lesser degree than points in the center of cluster. An overview and comparison of different fuzzy clustering algorithms is available.

Any point x has a set of coefficients giving the degree of being in the kth cluster wk(x). With fuzzy c-means, the centroid of a cluster is the mean of all points, weighted by their degree of belonging to the cluster:

$$c_k = \frac{\sum_x w_k(x) x}{\sum_x w_k(x)}.$$

The degree of belonging, wk(x), is related inversely to the distance from x to the cluster centrer as calculated on the previous pass. It also depends on a parameter m that controls how much weight is given to the closest centre. The fuzzy c-means algorithm is very similar to the k-means algorithm:

- Choose a number of clusters.
- Assign randomly to each point coefficients for being in the clusters.
- Repeat until the algorithm has converged (that is, the coefficients' change between two iterations is no more than ε, the given sensitivity threshold) :
  - Compute the centroid for each cluster, using the formula above.
  - For each point, compute its coefficients of being in the clusters, using the formula above.

The algorithm minimizes intra-cluster variance as well, but has the same problems as k-means; the minimum is a local minimum, and the results depend on the initial choice of weights.

The expectation-maximization algorithm is a more statistically formalized method which includes some of these ideas: partial membership in classes.

Fuzzy c-means has been a very important tool for image processing in clustering objects in an image. In the 70's, mathematicians introduced the spatial term into the FCM algorithm to improve the accuracy of clustering under noise.

QT clustering algorithm

QT (quality threshold) clustering (Heyer, Kruglyak, Yooseph 1999) is an alternative method of partitioning data, invented for gene clustering. It requires more computing power than k-means, but does not require specifying the number of clusters a priori, and always returns the same result for equal inputs.

The algorithm is:

- The user chooses a maximum diameter for clusters.
- Build a candidate cluster for each point by iteratively including the point that is closest to the group, until the diameter of the cluster surpasses the threshold.
- Save the candidate cluster with the most points as the first true cluster, and remove all points in the cluster from further consideration. Must clarify what happens if more than 1 cluster has the maximum number of points ?
- Recurse with the reduced set of points.

The distance between a point and a group of points is computed using complete linkage, i.e. as the maximum distance from the point to any member of the group (see Agglomerative hierarchical clustering, above, which describes various distance metrics between clusters).

Locality-sensitive hashing can be used for clustering. Feature space vectors are sets, and the metric used is the Jaccard distance. The feature space can be considered high-dimensional. The MinHash min-wise independent permutations LSH scheme is then used to put similar items into buckets. With just one set of hashing methods, there are only clusters of very similar elements. By seeding the hash functions several times (e.g. 20), it is possible to get bigger clusters.[6]

Formal concept analysis is a technique for generating clusters (called formal concepts) of objects and attributes, given a bipartite graph representing the relation between the objects and attributes. This technique was introduced by Rudolf Wille in 1984. Other methods for generating overlapping clusters (a cover rather than a partition) are discussed by Jardine and Sibson (1968) and Cole and Wishart (1970).

Spectral clustering

Given a set of data points A, the similarity matrix may be defined as a matrix S where Sij represents a measure of the similarity between points $i, j \in A$. Spectral clustering techniques make use of the spectrum of the similarity matrix of the data to perform dimensionality reduction for clustering in fewer dimensions.

One such technique is the Normalized Cuts algorithm or Shi–Malik algorithm introduced by Jianbo Shi and Jitendra Malik, commonly used for image segmentation. It partitions points into two sets (S1,S2) based on the eigenvector v corresponding to the second-smallest eigenvalue of the Laplacian matrix

$$L = I - D - 1 / 2SD - 1 / 2$$

of S, where D is the diagonal matrix

$$Di = \sum Sij.j$$

This partitioning may be done in various ways, such as by taking the median m of the components in v, and placing all points whose component in v is greater than m in S1, and the rest in S2. The algorithm can be used for hierarchical clustering by repeatedly partitioning the subsets in this fashion.

A related algorithm is the Meila–Shi algorithm (Marina Meilă & Jianbo Shi 2000), which takes the eigenvectors corresponding to the k largest eigenvalues of the matrix P = SD − 1 for some k, and then invokes another algorithm (e.g. k-means) to cluster points by their respective k components in these eigenvectors.

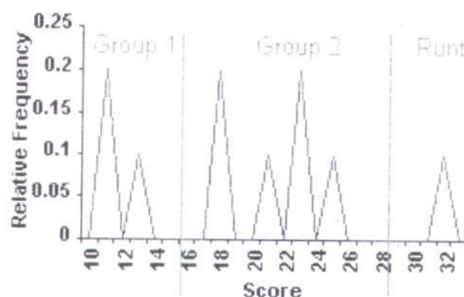
IV.3 Links and Dendogram

In cases of one or two measures, a visual inspection of the data using a frequency polygon or scatterplot often provides a clear picture of grouping possibilities. For example, the following is the data from the "Example Assignment" of the cluster analysis homework assignment.

Example Assignment

	Julie	John	Ryan	Bob	Ted	Kristi	Carol	Alice	Kari	Dave
SCORE	11	11	13	18	18	21	23	23	25	32

The relative frequency polygon appears as follows:



It is fairly clear from this picture that two subgroups, the first including Julie, John, and Ryan and the second including everyone else except Dave describe the data fairly well. When faced with complex multivariate data, such visualization procedures are not available and computer programs assist in assigning objects to groups. The following text describes the logic involved in cluster analysis algorithms.

Steps in Doing a Cluster Analysis

A common approach to doing a cluster analysis is to first create a table of relative similarities or differences between all objects and second to use this information to combine the objects into groups. The table of relative similarities is called a proximities matrix. The method of combining objects into groups is called a clustering algorithm. The idea is to combine objects that are similar to one another into separate groups.

The Proximities Matrix

Cluster analysis starts with a data matrix, where objects (usually people in the social sciences) are rows and observations are columns. From this beginning, a table is constructed where objects are both rows and columns and the numbers in the table are measures of similarity or differences between the two observations.

For example, given the following data matrix:

	X1	X2	X3	X4	X5
O1					
O2					
O3					
O4					

A proximities matrix would appear as follows:

	O1	O2	O3	O4
O1				
O2				
O3				
O4				

The difference between a proximities matrix in cluster analysis and a correlation matrix is that a correlation matrix contains similarities between variables (X1, X2) while the proximities matrix contains similarities between observations (O1, O2).

The researcher has dual problems at this point. The first is a decision about what variables to collect and include in the analysis. Selection of irrelevant measures will not aid in classification. For example, including the number of

legs an animal has would not help in differentiating cats and dogs, although it would be very valuable in differentiating between spiders and insects.

The second problem is how to combine multiple measures into a single number, the similarity between the two observations. This is the point where univariate and multivariate cluster analysis separate. Univariate cluster analysis groups are based on a single measure, while multivariate cluster analysis is based on multiple measures.

Univariate Measures

A simpler version of the problem of how to combine multiple measures into a measure of difference between objects is how to combine a single observation into a measure of difference between objects. Consider the following scores on a test for four students:

Student	Score
Julie	11
John	11
Ryan	13
Bob	18

The proximities matrix for these four students would appear as follows:

	Julie	John	Ryan	Bob
Julie				
John				
Ryan				
Bob				

The entries of this matrix will be described using a capital "D", for distance with a subscript describing which row and column. For example, D34 would describe the entry in row 3, column 4, or in this case, the intersection of Ryan and Bob.

One means of filling in the proximities matrix is to compute the absolute value of the difference between scores. For example, the distance, D34, between Ryan and Bob would be |13-18| or 5. Completing the proximities matrix using the example data would result in the following:

	Julie	John	Ryan	Bob
Julie	0	0	2	7
John	0	0	2	7
Ryan	2	2	0	5
Bob	7	7	5	0

A second means of completing the proximities matrix is to use the squared difference between the two measures. Using the example above, D34, the distance between Ryan and Bob, would be (13-18)2 or 25. This distance measure has the advantage of being consistent with many other statistical measures, such as variance and the least squares criterion, and will be used in the examples that follow. The example proximities matrix using squared differences as the distance measure is presented below.

	Julie	John	Ryan	Bob
Julie	0	0	4	49
John	0	0	4	49
Ryan	4	4	0	25
Bob	49	49	25	0

Note that both example proximities matrices are symmetrical. Symmetrical means that row and column entries can be interchanged or that the numbers are the same on each half of the matrix defined by a diagonal running from top left to bottom right.

Multivariate Measures

When more than one measure is obtained for each observation, then some method of combining the proximities matrices for different measures must be found. Usually the matrices are summed in a combined matrix. For example, given the following scores.

	X1	X2
O1	25	11
O2	33	11
O3	34	13
O4	35	18

The two proximities matrices resulting from squared Euclidean distance that result could be summed to produce a combined distance matrix.

	O1	O2	O3	O4
O1	0	64	81	100
O2	64	0	1	4
O3	81	1	0	1
4	100	4	1	0

+

	O1	O2	O3	O4
O1	0	0	4	49
O2	0	0	4	49
O3	4	4	0	25
O4	49	49	25	0

=

	O1	O2	O3	O4
O1	0	64	85	149
O2	64	0	5	53
O3	85	5	0	26
O4	149	53	26	0

Note that each corresponding cell is added. With more measures there are more matrices to be added together.

This system works reasonably well if the measures share similar scales. One measure can overwhelm the other if the measures use different scales. Consider the following scores.

	X1	X2
O1	25	11
O2	33	21
O3	34	33
O4	35	48

The two proximities matrices resulting from squared Euclidean distance that result could be summed to produce a combined distance matrix.

	O1	O2	O3	O4
O1	0	64	81	100
O2	64	0	1	4
O3	81	1	0	1
O4	100	4	1	0

+

	O1	O2	O3	O4
O1	0	100	484	49
O2	100	0	144	729
O3	484	144	0	225
O4	1369	729	225	0

=

	O1	O2	O3	O4
O1	0	164	485	153
O2	164	0	145	733
O3	565	145	0	226
O4	1469	733	226	0

It can be seen that the second measure overwhelms the first in the combined matrix. For this reason the measures are optionally transformed before they are combined. For example, the previous data matrix might be converted to standard scores before computing the separated distance matrices.

	X1	X2	Z1	Z2
O1	25	11	-1.48	-1.08
O2	33	21	.27	-.45
O3	34	33	.49	.30
O4	35	48	.71	1.24

The two proximities matrices resulting from squared Euclidean distance that result from the standard scores could be summed to produce a combined distance matrix.

	O1	O2	O3	O4
O1	0	3.06	3.88	4.80
O2	3.06	0	.05	.19
O3	3.88	.05	0	.05
O4	4.80	.19	.05	0

+

	O1	O2	O3	O4
O1	0	.40	1.90	5.38
O2	.40	0	.56	2.86
O3	1.9	.56	0	.88
O4	5.38	2.86	.88	0

=

	O1	O2	O3	O4
O1	0	3.46	5.78	10.18
O2	3.46	0	.61	3.05
O3	5.78	.61	0	.93
O4	10.18	3.05	.93	0

The point is that the choice of whether to transform the data and the choice of distance metric can result in vastly different proximities matrices.

Using Distances to Group Objects

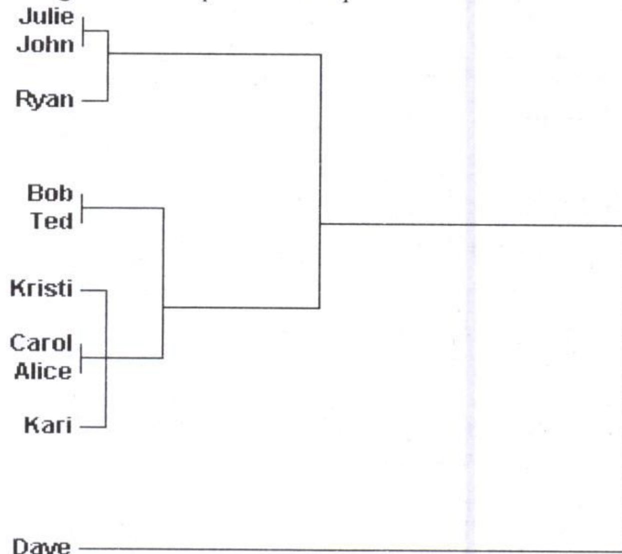
After the distances between objects have been found, the next step in the cluster analysis procedure is to divide the objects into groups based on the distances. Again, any number of options is available to do this.

If the number of groups is known beforehand, a "flat" method might be preferable. Using this method, the objects are assigned to a given group at the first step based on some initial criterion. The means for each group are calculated. The next step reshuffles the objects into groups, assigning objects to groups based on the object's similarity to the current mean of that group. The means of the groups are recalculated at the end of this step. This process continues recursively until no objects change groups. This idea is the basis for "k-means cluster analysis" available on SPSS/WIN and other statistical packages. This method works well if the number of groups matches the data and the initial solution is reasonably close to the final solution.

Hierarchical clustering methods do not require preset knowledge of the number of groups. Two general methods of hierarchical clustering methods are available: divisive and agglormerative.

The divisive techniques start by assuming a single group, partitioning that group into subgroups, partitioning these subgroups further into subgroups and so on until each object forms its own subgroup. The agglormerative techniques start with each object describing a subgroup, and then combine like subgroups into more inclusive subgroups until only one group remains. The agglormerative techniques will be described further in this chapter, although many of the procedures described would hold for either method.

In either case, the results of the application of the clustering technique are best described using a dendogram or binary tree. The objects are represented as nodes in the dendogram and the branches illustrate when the cluster method joins subgroups containing that object. The length of the branch indicates the distance between the subgroups when they are joined. An example dendogram using the example data is presented below:



The interpretation of a dendogram is fairly straightforward. In the above dendogram, for example, Julie, John, and Ryan form a group, Bob, Ted, Kristi, Carol, Alice, and Kari form a second group, and Dave is called a "runt" because he doesn't enter any group until near the end of the procedure. Different methods exist for computing the distance between subgroups at each step in the clustering algorithm. Again, statistical packages give options for which procedure to use.

The following methods will now be discussed: single linkage or nearest neighbor, complete linkage or furthest neighbor, and average linkage.

Single Linkage

Single linkage (nearest neighbor in SPSS/WIN) computes the distance between two subgroups as the minimum distance between any two members of opposite groups. For example, consider the following proximities matrix.

	Julie	John	Ryan	Bob	Ted	Kristi
Julie	0	4	36	81	196	225
John	4	0	16	49	144	169
Ryan	36	16	0	9	64	81
Bob	81	49	9	0	25	36
Ted	196	144	64	25	0	1
Kristi	225	169	81	36	1	0

Based on this data, Ted and Kristi would be joined on the first step.

	Julie	John	Ryan	Bob	Ted and Kristi
Julie	0	4	36	81	196
John	4	0	16	49	144
Ryan	36	16	0	9	64
Bob	81	49	9	0	25
Ted and Kristi	196	144	64	25	0

Julie and John would be joined on the second step.

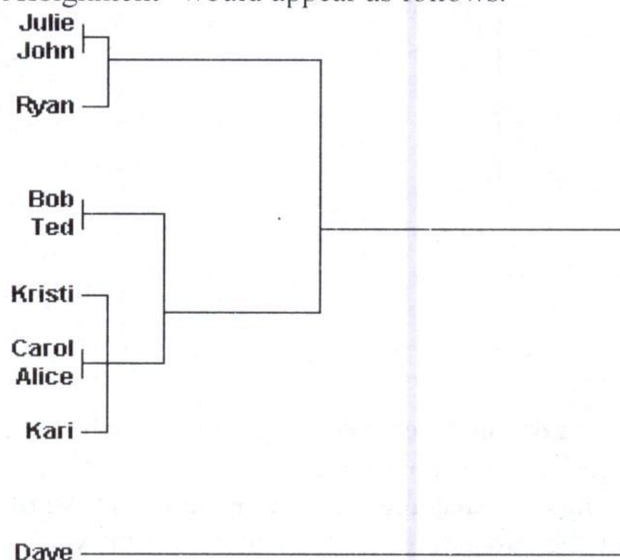
	Julie and John	Ryan	Bob	Ted and Kristi
Julie and John	0	16	49	144
Ryan	16	0	9	64
Bob	49	9	0	25
Ted and Kristi	144	64	25	0

Ryan and Bob would be joined on the third step. At that step four would be three groups of two each and the proximities matrix at that point would appear as follows.

	Julie and John	Ryan and Bob	Ted and Kristi
Julie and John	0	16	144
Ryan and Bob	16	0	25
Ted and Kristi	144	25	0

Using single linkage, Julie and John would be grouped with Ryan and Bob at the third step. The distance of this linkage would be 16. The last step would join all into a single group with a distance of 25.

The single linkage dendogram of the data generated by "Example Assignment" would appear as follows.



Complete Linkage

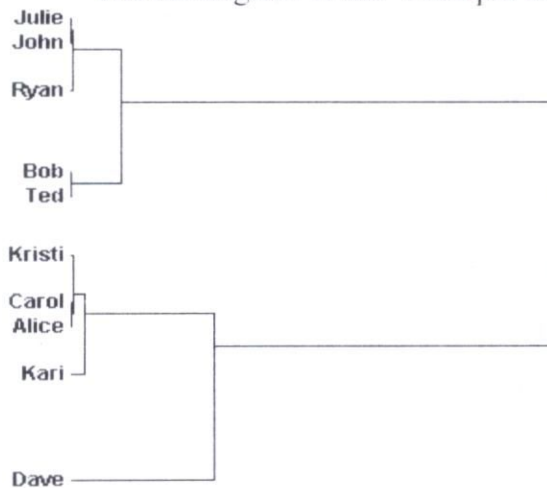
Complete linkage (furthest neighbor in SPSS/WIN) computes the distance between subgroups in each step as the maximum distance between any two members of the different groups. Using the example proximities matrix, complete linkage would join similar members at steps 1, 2, and 3 in the

procedure. At step four the proximities matrix for the three groups would appear as follows.

	Julie and John	Ryan and Bob	Ted and Kristi
Julie and John	0	81	225
Ryan and Bob	81	0	81
Ted and Kristi	225	81	0

Step four would combine all three subgroups into a single group with a distance of 81.

The dendogram of the "example assignment" data is presented below.

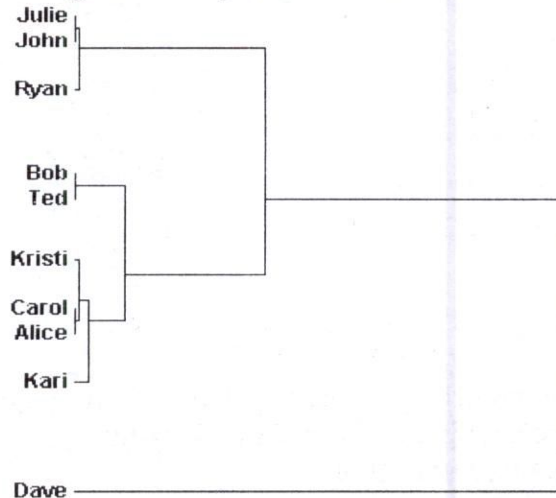


Average Linkage

Average linkage computes the distance between subgroups at each step as the average of the distances between the two subgroups.
Using the example proximities matrix, average linkage would join similar members at steps 1, 2, and 3 in the procedure. At step four, the proximities matrix for the three groups would appear as follows.

	Julie and John	Ryan and Bob	Ted and Kristi
Julie and John	0	45.5	51.5
Ryan and Bob	45.5	0	81
Ted and Kristi	183.5	51.5	0

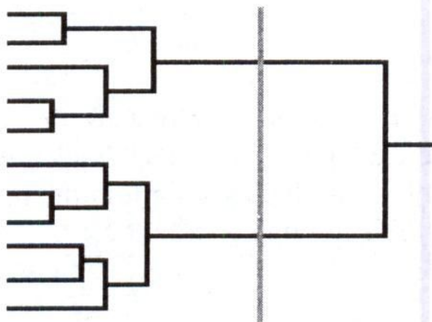
Thus, Julie and John would be grouped with Ryan and Bob with a distance of 51.5. The final step would join this group with Ted and Kristi with an average distance of 117.5. The dendogram for average linkage in the "Example Assignment" is presented below.



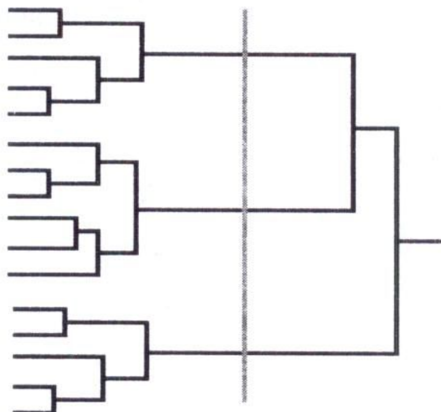
A program generating a cluster analysis homework assignment permits the student to view a dendogram for single, complete, and average linkage for a random proximities matrix. The student should verify that different dendograms result when different linkage methods are used.

Interpreting a Dendogram

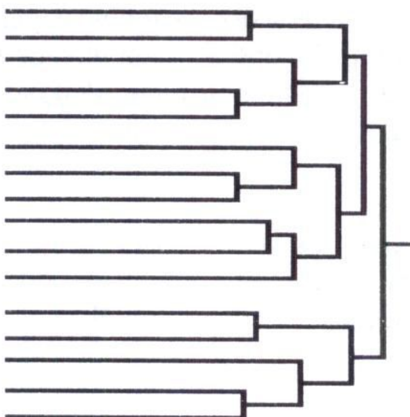
A dendogram that clearly differentiates groups of objects will have small distances in the far branches of the tree and large differences in the near branches. The following example dendogram ideally illustrates two clear groups.



The following dendogram ideally illustrates a clear grouping of three groups.



When the distances on the far branches are large relative to the near branches, then the grouping is not very effective. The following illustrates a dendogram that would be interpreted with great caution.



Dendograms are also useful in discovering "runts", or objects that are joined to a group in the near branches. In the example data "Dave" is clearly a runt, at least in the single linkage dendogram, because he does not join the main group until the last step. Runts are exceptions to the grouping structure.

It has been suggested (Stuetzle, 1995) that some statisticians prefer complete linkage because deceptively cleaner dendograms are often produced. A unimodal distribution will often produce a clean dendogram that might be interpreted as two groups even though no grouping is appropriate. For this reason Stuetzle (1995) recommends that single linkage be used and closely examined for runts.

Conclusion and Cautions

Cluster analysis methods will always produce a grouping. The groupings produced by cluster analysis may or may not prove useful for classifying objects. If the groupings discriminate between variables not used to do the grouping and those discriminations are useful, then cluster analysis is useful. For example, if grouping zip code areas into fifteen categories based on age, gender, education, and income discriminates between wine drinking behaviors, it would be very useful information if one was interested in expanding a wine store into new areas.

Cluster analysis may be used in conjunction with discriminant function analysis. After multivariate data are collected, observations are grouped using cluster analysis. Discriminant function analysis is then used on the resulting groups to discover the linear structure of either the measures used in the cluster analysis and/or different measures.

Cluster analysis methods are not clearly established. There are many options one may select when doing a cluster analysis using a statistical package. Cluster analysis is thus open to the criticism that a statistician may mine the data trying different methods of computing the proximities matrix and linking groups until he or she "discovers" the structure that he or she originally believed was contained in the data. One wonders why anyone would bother to do a cluster analysis for such a purpose.

IV.4 Partitional Algorithm

Clustering is an unsupervised machine learning algorithm that groups entities, from a dataset, that have high degree of similarity in a same cluster.

Nowadays lots of areas are using this kind of algorithms to separate datasets into groups in a automated way, and still have a good quality result.

The clustering process is not a universal process because that are a lot of kind of groups of datasets, for some of this the kind of metric is a relevant thing, for others the entities that represent each cluster are more interesting. Like datasets groups there are a lot of kind of clustering algorithms each one tries to take advantage of a kind of group of data, so this way each one of them is more suited to a more specific kind of data.

MST based clustering algorithm

The basic idea of MST based clustering algorithm is as follows.

First construct MST(minimum spanning tree) using Kruskal algorithm and then set a threshold value and step size. We then remove those edges from the MST, whose lengths are greater than the threshold value. We next calculate the ratio between the intra-cluster distance and inter-cluster distance and record the ratio as well as the threshold. We update the threshold value by incrementing the step size. Every time we obtain the new (updated) threshold value, we repeat the above procedure. We stop repeating, when we encounter a situation such that the threshold value is maximum and as such no MST edges can be removed. In such situation, all the data points belong to a single cluster. Finally we obtain the minimum value of the recorded ratio and form the clusters corresponding to the stored threshold value. The above algorithm has two extreme cases:

1) With the zero threshold value, each point remains within a single cluster.
2) With the maximum threshold value all the points lie within a single cluster.

Therefore, the proposed algorithm searches for that optimum value of the threshold for which the Intra-Inter distance ratio is minimum. It needs not to mention that this optimum value of the threshold must lie between these two extreme values of the threshold. However, in order to reduce the number of iteration we never set the initial threshold value to zero.

Advantage
1) Comparatively better performance then k-means algorithm.

Disadvantage
1) Threshold value and step size needs to be defined apriori

Square error clustering methods

The most commonly used clustering strategy is based on the square-root error criterion.

Objective: To obtain a partition which, for a fixed number of clusters, minimizes the square-error where square-error is the sum of the Euclidean distances between each pattern and its cluster center.

Algorithm

1. Select an initial partition with k clusters. Repeat steps 2 through 5 until the cluster membership stabilizes.
2. Generate a new partition by assigning each pattern to its closest cluster center.
3. Compute new cluster centers as the centroids of the clusters.
4. Repeat steps 2 and 3 until an optimum value of the criterion is found.
5. Adjust the number of clusters by merging and splitting existing clusters or by removing small or outlier clusters.

The algorithm converges when the criterion function cannot be improved.

Initial partition

- Select k seed points at random or by taking the centroid as the first seed point and the rest at a certain minimum distance from this seed point.
- Cluster the remaining points to the closest seed point.

Updating a partition

K-means:

- In each pass(cycle) make an assignment of all patterns to the closest cluster center.
- Recompute the cluster center after every new assignment is made.

Adjusting the number of clusters

- Clustering algorithms can create new clusters or merge existing ones if certain conditions specified by the user are met.

- Split a cluster if it has too many patterns and an unusually large variance along the feature with large spread.
- Merge if they are sufficiently close.
- Remove outliers from future consideration. (outliers are pattern/patterns that is sufficiently far removed from the rest of the data and hence suspected as a mistake in data entry.)

Performance of square-error clustering methods

- Seeks compact hyper-ellipsoidal clusters and this can produce misleading results when the data do not occur in compact, hyper-ellipsoidal boundaries.
- Exhibit inadequacies when the Euclidean measure is used to measure distance but the features are not on comparable scales.

K-Means Clustering

K-means is one of the simplest unsupervised learning algorithms that solve the well known clustering problem. The procedure follows a simple and easy way to classify a given data set through a certain number of clusters (assume k clusters) fixed a priori. The main idea is to define k centroids, one for each cluster. These centroids should be placed in a cunning way because of different location causes different result. So, the better choice is to place them as much as possible far away from each other. The next step is to take each point belonging to a given data set and associate it to the nearest centroid. When no point is pending, the first step is completed and an early groupage is done. At this point we need to re-calculate k new centroids as barycenters of the clusters resulting from the previous step. After we have these k new centroids, a new binding has to be done between the same data set points and the nearest new centroid. A loop has been generated. As a result of this loop we may notice that the k centroids change their location step by step until no more changes are done. In other words centroids do not move any more. Finally, this algorithm aims at minimizing an objective function, in this case a squared error function.

The objective function

$$J = \sum_{j=1}^{k} \sum_{i=1}^{n} \left\| x_i^{(j)} - c_j \right\|^2$$

where $\left\|x_i^{(j)} - c_j\right\|^2$ is a chosen distance measure between a data point $x_i^{(j)}$ and the cluster centre c_j, is an indicator of the distance of the n data points from their respective cluster centres.

The algorithm is composed of the following steps:

1. Place K points into the space represented by the objects that are being clustered. These points represent initial group centroids.
2. Assign each object to the group that has the closest centroid.
3. When all objects have been assigned, recalculate the positions of the K centroids.
4. Repeat Steps 2 and 3 until the centroids no longer move. This produces a separation of the objects into groups from which the metric to be minimized can be calculated.

Although it can be proved that the procedure will always terminate, the k-means algorithm does not necessarily find the most optimal configuration, corresponding to the global objective function minimum. The algorithm is also significantly sensitive to the initial randomly selected cluster centres. The k-means algorithm can be run multiple times to reduce this effect.

K-means is a simple algorithm that has been adapted to many problem domains. As we are going to see, it is a good candidate for extension to work with fuzzy feature vectors.

An example

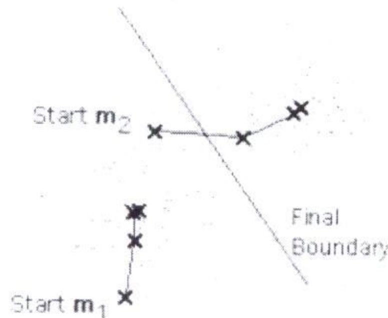
Suppose that we have n sample feature vectors x1, x2, ... xn all from the same class, and we know that they fall into k compact clusters, k < n. Let mi be the mean of the vectors in cluster i. If the clusters are well separated, we can use a minimum-distance classifier to separate them. That is, we can say that x is in cluster i if || x - mi || is the minimum of all the k distances. This suggests the following procedure for finding the k means:

- Make initial guesses for the means m1, m2, ..., mk
- Until there are no changes in any mean

- Use the estimated means to classify the samples into clusters
- For i from 1 to k
  - Replace mi with the mean of all of the samples for cluster i
- end_for

- end_until

Here is an example showing how the means m1 and m2 move into the centers of two clusters.

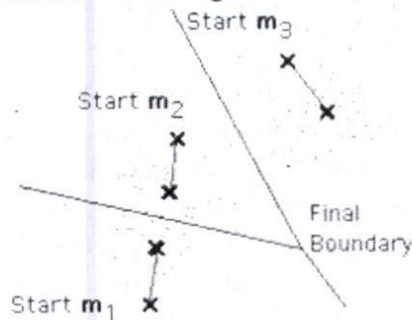


Remarks

This is a simple version of the k-means procedure. It can be viewed as a greedy algorithm for partitioning the n samples into k clusters so as to minimize the sum of the squared distances to the cluster centers. It does have some weaknesses:

- The way to initialize the means was not specified. One popular way to start is to randomly choose k of the samples.
- The results produced depend on the initial values for the means, and it frequently happens that suboptimal partitions are found. The standard solution is to try a number of different starting points.
- It can happen that the set of samples closest to mi is empty, so that mi cannot be updated. This is an annoyance that must be handled in an implementation, but that we shall ignore.
- The results depend on the metric used to measure || x - mi ||. A popular solution is to normalize each variable by its standard deviation, though this is not always desirable.
- The results depend on the value of k.

This last problem is particularly troublesome, since we often have no way of knowing how many clusters exist. In the example shown above, the same algorithm applied to the same data produces the following 3-means clustering. Is it better or worse than the 2-means clustering?



Unfortunately there is no general theoretical solution to find the optimal number of clusters for any given data set. A simple approach is to compare the results of multiple runs with different k classes and choose the best one according to a given criterion (for instance the Schwarz Criterion - see Moore's slides), but we need to be careful because increasing k results in smaller error function values by definition, but also an increasing risk of over fitting.

The Partitioning Around Medoids (PAM) Algorithm
Algorithm

The PAM algorithm was developed by Leonard Kaufman and Peter J. Rousseeuw, and this algorithm is very similar to K-means, mostly because both are partitional algorithms, in other words, both break the datasets into groups, and both works trying to minimize the error, but PAM works with Medoids, that are an entity of the dataset that represent the group in which it is inserted, and K-means works with Centroids, that are artificially created entity that represent its cluster.

The PAM algorithm partitionates a dataset of n objects into a number k of clusters, where both the dataset and the number k is an input of the algorithm. This algorithm works with a matrix of dissimilarity, where its goal is to minimize the overall dissimilarity between the representants of each cluster and its members. The algorithm uses the following model to solve the problem:

$F(x) = \text{minimize} \sum_{i=1}^{n} \sum_{j=1}^{n} d(i,j) z_{ij}$

Subject to:

1. $\Sigma_{i=1}^{n} z_{ij} = 1$, $j = 1,2,...,n$
2. $z_{ij} \leq y_i$, $i, j = 1,2,...,n$
3. $\Sigma_{i=1}^{n} y_i = k$, k = number of clusters
4. y_i , $z_{ij} \in \{0,1\}$, $i, j = 1,2,...,n$

Where F(x) is the main function to minimize, where d(i,j) is the dissimilarity measurement between the entities i and j, and z_{ij} is a variable that ensures that only the dissimilarity between entities from the same cluster will be computed in the main function. The others expressions are constraints that have the following functions: (1.) ensures that every single entity is assigned to one cluster and only one cluster, (2.) ensures that the entity is assigned to its medoid that represent the cluster, (3.) ensures that there is a number exactly equals to k of clusters and (4.) let the decision variables assume just the values of 0 or 1.

The PAM algorithm can work over two kind of input, the first is the matrix representing every entity and the values of its variables, and the second is to work with the dissimilarity matrix directly, in the later the user can provide the dissimilarity directly as an input to the algorithm, instead of the data matrix representing the entities. Either way the algorithm reach an solution to the problem, in a general analysis the algorithm proceed this way:

Build phase:

1. Choose k entities to become the medoids, or in case these entities were provided use them as the medoids;
2. Calculate the dissimilarity matrix if it was not informed;
3. Assign every entity to its closest medoid;

Swap phase:

1. For each cluster search if any of the entities of the cluster lower the average dissimilarity coefficient, if it does select the entity that lower the most this coefficient as the medoid for this cluster;
2. If at least the medoid from one cluster has changed go to (3), else end the algorithm.

As was said the PAM algorithm works with a matrix of dissimilarity, and to calculate this matrix the algorithm can use two metrics the first one is the

euclidean, that are the root sum-of-squares of differences, while the second one is the manhattan distance that are the sum of absolute distances.

Implemantation

The pseudocode of PAM algorithm is shown below:

Algorithm 1: PAM Algorithm Input: E = {e1,e2,...,en} (dataset to be clustered or matrix of dissimilarity)

k (number of clusters)

metric (kind of metric to use on dissimilarity matrix)

diss (flag indicating that E is the matrix of dissimilarity or not)

Output: M = {m1,m2,....mk} (vector of clusters medoids)

L = {l(e) | e = 1,2,...,n} (set of cluster labels of E)

foreach m_i € M **do**

$m_i \leftarrow e_j$ € E: (e.g. random selection)

end if diss ≠ true

Dissimilarity ← CalculateDissimilarityMatrix(E, metric);

else

Dissimilarity ← E;

end repeat

foreach e_i € E **do**

$l(e_i)$ ← argminDissimilarity(e_i, Dissimilarity, M);

end

changed ← false;

foreach m_i € M **do**

Mtmp ← SelectBestClusterMedoids(E, Dissimilarity, L);

end

if Mtmp ≠ M

M ← Mtmp;

changed ← true;

end

until changed = true;

In the R programming language, the PAM algorithm is available in the cluster package and can be called by the following command:

pam(x, k, diss, metric, medoids, stand, cluster.only, do.swap, keep.diss, keep.data,
trace.lev)

Where the parameters are:

x: numerical data matrix representing the dataset entities, or can be the dissimilarity matrix, it depends on the value of the diss parameter. In case x is a data matrix each row is an entity and each column is an variable, and in this case missing values are allowed as long as every pair of entities has at least one case not missing. In case x is a dissimilarity matrix it is not allowed to have missing values.

k: number of clusters that the dataset will be partitioned where $0 < k < n$, where n is the number of entities.

diss: logical flag, if it is TRUE x is used as the dissimilarity matrix, if it is FALSE, then x will be considered as a data matrix.

metric: an string specifying each of the two metrics will be used to calculate the dissimilarity matrix, the metric variable can be "euclidean" to use the Euclidean distance, or can be "manhattan" to use the Manhattan distance.

stand: logical flag, if it is TRUE then the measurements in x will be standardized before calculating the dissimilarities. Measurements are standardized for each column, by subtracting the column's mean value and dividing by the variable's mean absolute deviation. If x is a dissimilarity matrix then this parameter is ignored.

cluster.only: logical flag, if it is TRUE, only the clustering will be computed and returned. do.swap logical flag, indicates if the swap phase should happen (TRUE) or not (FALSE).

keep.diss: logical flag indicating if the dissimilarities should (TRUE) or not (FALSE) be kept in the result.

keep.data: logical flag indicating if the input data x should (TRUE) or not (FALSE) be kept in the result.

trace.lev: an numeric parameters specifying a trace level for printing diagnostics during the build and swap phase of the algorithm. Default 0 does not print anything.

The PAM algorithm return a pam object that contains the information about the result of the execution of the algorithm.

Genetic Algorithm

Finally, the third approach is based on the use of Genetic Algorithms (GA). Evolution has proven to be a very powerful mechanism in finding good solutions to difficult problems. One can look at the natural selection as an optimisation method, which tries to produce adequate solutions to particular environments.

In spite of the large number of applications of GA in different types of optimisation problems, there is very little research on using this kind of approach to the clustering problem. In fact, and bearing in mind the quality of the solutions that this technology has showed in different types of fields and problems (Beasley, Bull and Martin, 1993a, Mitchell, 1996) it makes perfect sense to try to use it in clustering problems.

The flexibility associated with GA is one important aspect to bear in mind. With the same genome representation and just by changing the fitness function one can have a different algorithm. In the case of spatial analysis this is particularly important since one can try different fitness functions in an exploratory phase.

The GA minimizes the square error of the cluster dispersion:

$$E = \sum_{k=1}^{K} \sum_{x \in Ck} \left\| x - mk \right\|^2$$

K being the number of clusters, mk the centre of cluster Ck, which makes it similar to the k-means algorithm.

In the genome each gene represents a data point and defines cluster membership. All necessary evolution operators can be implemented with this scheme. As pointed by Demiriz et all (1999) the major problem associated with this representation scheme is that it is not scalable, on the other hand it seems to be computationally efficient when the number of data points is not too large. The GA was implemented through the customisation of Galib (1996). The population size used was 100, with a replacement percentage of 50%, a crossover probability of 0.9 and a mutation probability of 0.01, uniform crossover was applied and 15000 generations were evaluated.

IV.5 Categorical Algorithm

Clustering Algorithms for Categorical Data Sets

- As mentioned earlier, one essential issue for clustering a categorical data set is to define a similarity (dissimilarity) function between two objects.
- One of the most fundamental and important data models of categorical data sets is the market-basket data model.

IV.6 Summary

Cluster analysis classifies a set of observations into two or more mutually exclusive unknown groups based on combinations of interval variables. The purpose of cluster analysis is to discover a system of organizing observations, usually people, into groups.

Hierarchical clustering creates a hierarchy of clusters which may be represented in a tree structure called a dendrogram. The root of the tree consists of a single cluster containing all observations, and the leaves correspond to individual observations.

Algorithms for hierarchical clustering are generally either agglomerative, in which one starts at the leaves and successively merges clusters together; or divisive, in which one starts at the root and recursively splits the clusters.

Any non-negative-valued function may be used as a measure of similarity between pairs of observations. The choice of which clusters to merge or split is determined by a linkage criterion, which is a function of the pair-wise distances between observations.

The maximum distance between elements of each cluster (also called complete-linkage clustering):

$$\max\{\, d(x,y) : x \in \mathcal{A},\, y \in \mathcal{B}\,\}.$$

The minimum distance between elements of each cluster (also called single-linkage clustering):

$$\min\{\, d(x,y) : x \in \mathcal{A},\, y \in \mathcal{B}\,\}.$$

The mean distance between elements of each cluster (also called average linkage clustering, used e.g. in UPGMA):

$$\frac{1}{|\mathcal{A}|\cdot|\mathcal{B}|}\sum_{x\in\mathcal{A}}\sum_{y\in\mathcal{B}} d(x,y).$$

In fuzzy clustering, each point has a degree of belonging to clusters, as in fuzzy logic, rather than belonging completely to just one cluster.

The fuzzy c-means algorithm is very similar to the k-means algorithm:

- Choose a number of clusters.
- Assign randomly to each point coefficients for being in the clusters.
- Repeat until the algorithm has converged (that is, the coefficients' change between two iterations is no more than ε, the given sensitivity threshold) :
  - Compute the centroid for each cluster, using the formula above.

- For each point, compute its coefficients of being in the clusters, using the formula above.
-

QT (quality threshold) clustering (Heyer, Kruglyak, Yooseph 1999) is an alternative method of partitioning data, invented for gene clustering. It requires more computing power than k-means, but does not require specifying the number of clusters a priori, and always returns the same result for equal inputs. The algorithm is:

- The user chooses a maximum diameter for clusters.
- Build a candidate cluster for each point by iteratively including the point that is closest to the group, until the diameter of the cluster surpasses the threshold.
- Save the candidate cluster with the most points as the first true cluster, and remove all points in the cluster from further consideration. Must clarify what happens if more than 1 cluster has the maximum number of points ?
- Recurse with the reduced set of points.

K-means is one of the simplest unsupervised learning algorithms that solve the well known clustering problem. The procedure follows a simple and easy way to classify a given data set through a certain number of clusters (assume k clusters) fixed a priori. The main idea is to define k centroids, one for each cluster. These centroids should be placed in a cunning way because of different location causes different result.

The algorithm is composed of the following steps:

1. Place K points into the space represented by the objects that are being clustered. These points represent initial group centroids.
2. Assign each object to the group that has the closest centroid.
3. When all objects have been assigned, recalculate the positions of the K centroids.
4. Repeat Steps 2 and 3 until the centroids no longer move. This produces a separation of the objects into groups from which the metric to be minimized can be calculated.

Algorithm : PAM Algorithm Input: E = {e1,e2....en} (dataset to be clustered or matrix of dissimilarity)

k (number of clusters)

metric (kind of metric to use on dissimilarity matrix)

diss (flag indicating that E is the matrix of dissimilarity or not)

Output: M = {m1,m2,....,mk} (vector of clusters medoids)

L = {l(e) | e = 1,2,....n} (set of cluster labels of E)

foreach m_i € M **do**

$m_i \leftarrow e_j$ € E; (e.g. random selection)

end if diss ≠ true

Dissimilarity ← CalculateDissimilarityMatrix(E, metric);

else

Dissimilarity ← E;

end repeat

foreach e_i € E **do**

$l(e_i)$ ← argminDissimilarity(e_i, Dissimilarity, M);

end

changed ← false;

foreach m_i € M **do**

Mtmp ← SelectBestClusterMedoids(E, Dissimilarity, L);

end

if Mtmp ≠ M

M ← Mtmp;

changed ← true;

end

until changed = true;

The GA minimizes the square error of the cluster dispersion:

$$E = \sum_{k=1}^{K} \sum_{x \in C_k} \left\| x - m_k \right\|^2$$

UNIT V WEB MINING

Structure

V.0 Objectives

In this unit we are going to discuss about the basic concepts related to web mining. This unit also elaborates how web mining acts as the core process of knowledge discovery from web pages. It also discusses about ontology and web mining applications.

At the end of this unit we can

→ Understand the concept of web mining
→ Identify the kinds of web mining
→ Gain knowledge about the ontology
→ Know the applications of web mining

V.1 Web mining

With the explosive growth of information sources available on the World Wide Web, it has become increasingly necessary for users to utilize automated tools in find the desired information resources, and to track and analyze their usage patterns. These factors give rise to the necessity of creating server side and client side intelligent systems that can effectively mine for knowledge. Web mining can be broadly defined as the discovery and analysis of useful information from the World Wide Web. This describes the automatic search of information resources available online, i.e. Web content mining, and the discovery of user access patterns from Web servers, i.e., Web usage mining.

What is Web Mining ?

Web Mining is the extraction of interesting and potentially useful patterns and implicit information from artifacts or activity related to the World Wide Web. There are roughly three knowledge discovery domains that pertain to web mining: Web Content Mining, Web Structure Mining, and Web Usage Mining. Web content mining is the process of extracting knowledge from the content of documents or their descriptions. Web document text mining, resource discovery based on concepts indexing or agent based technology may also fall in this category. Web structure mining is the process of inferring knowledge from the World Wide Web organization and links between references and referents in the Web. Finally, web usage mining, also known as Web Log Mining, is the process of extracting interesting patterns in web access logs.

V.2 Web Content mining and usage mining

Web Content Mining

Web content mining is an automatic process that goes beyond keyword extraction. Since the content of a text document presents no machine readable semantic, some approaches have suggested restructuring the document content in a representation that could be exploited by machines. The usual approach to exploit known structure in documents is to use wrappers to map documents to some data model. Techniques using lexicons for content interpretation are yet to come.

There are two groups of web content mining strategies: Those that directly mine the content of documents and those that improve on the content search of other tools like search engines.

Web Structure Mining

World Wide Web can reveal more information than just the information contained in documents. For example, links pointing to a document indicate the popularity of the document, while links coming out of a document indicate the richness or perhaps the variety of topics covered in the document. This can be compared to bibliographical citations. When a paper is cited often, it ought to be

important. The PageRank and CLEVER methods take advantage of this information conveyed by the links to find pertinent web pages. By means of counters, higher levels cumulate the number of artifacts subsumed by the concepts they hold. Counters of hyperlinks, in and out documents, retrace the structure of the web artifacts summarized.

Web Usage Mining

Web servers record and accumulate data about user interactions whenever requests for resources are received. Analyzing the web access logs of different web sites can help understand the user behaviour and the web structure, thereby improving the design of this colossal collection of resources. There are two main tendencies in Web Usage Mining driven by the applications of the discoveries: General Access Pattern Tracking and Customized Usage Tracking. The general access pattern tracking analyzes the web logs to understand access patterns and trends. These analyses can shed light on better structure and grouping of resource providers. Many web analysis tools existd but they are limited and usually unsatisfactory. We have designed a web log data mining tool, WebLogMiner, and proposed techniques for using data mining and OnLine Analytical Processing (OLAP) on treated and transformed web access files. Applying data mining techniques on access logs unveils interesting access patterns that can be used to restructure sites in a more efficient grouping, pinpoint effective advertising locations, and target specific users for specific selling ads.

Customized usage tracking analyzes individual trends. Its purpose is to customize web sites to users. The information displayed, the depth of the site structure and the format of the resources can all be dynamically customized for each user over time based on their access patterns. While it is encouraging and exciting to see the various potential applications of web log file analysis, it is important to know that the success of such applications depends on what and how much valid and reliable knowledge one can discover from the large raw log data. Current web servers store limited information about the accesses. Some scripts custom tailored for some sites may store additional information. However, for an effective web usage mining, an important cleaning and data transformation step before analysis may be needed.

V.3 Ontology based web mining

The subject of ontology is the study of the categories of things that exist or may exist in some domain. The product of such a study, called an ontology, is a catalog of the types of things that are assumed to exist in a domain of interest D from the perspective of a person who uses a language L for the purpose of talking about D. The types in the ontology represent the predicates, word senses, or concept and relation types of the language L when used to discuss topics in the domain D. An un-interpreted logic, such as predicate calculus, conceptual graphs, or KIF, is ontologically neutral. It imposes no constraints on the subject matter or the way the subject may be characterized. By itself, logic says nothing about anything, but the combination of logic with an ontology provides a language that can express relationships about the entities in the domain of interest.

An informal ontology may be specified by a catalog of types that are either undefined or defined only by statements in a natural language. A formal ontology is specified by a collection of names for concept and relation types organized in a partial ordering by the type-subtype relation. Formal ontologies are further distinguished by the way the subtypes are distinguished from their super types: an axiomatized ontology distinguishes subtypes by axioms and definitions stated in a formal language, such as logic or some computer-oriented notation that can be translated to logic; a prototype-based ontology distinguishes subtypes by a comparison with a typical member or prototype for each subtype. Large ontologies often use a mixture of definitional methods: formal axioms and definitions are used for the terms in mathematics, physics, and engineering; and prototypes are used for plants, animals, and common household items.

V.4 Web mining Applications

The applications of web mining are listed below

- Web Mining Application in E-commerce Customer Behaviour Analysis
- Web Mining Application in E-commerce Transaction Analysis
- Web Mining Application in E-commerce Website Design
- Web Mining Application in E-banking
- Web Mining Application in M-commerce
- Web Mining Application in Web Advertisement
- Web Mining Application in Search Engine

- Web Mining Application in Online Auction
- Web Mining Application in Online Knowledge Management
- Web Mining Application in Online Social Networking
- Web Mining Application in E-learning
- Web Mining Application in Blog Analysis
- Web Mining Application in Online Personalization and
- Recommendation Systems
- Web Mining and Intelligent Web Services
- Case and Empirical Studies

V.5 Summary

Web mining can be broadly defined as the discovery and analysis of useful information from the World Wide Web. This describes the automatic search of information resources available online, i.e. Web content mining, and the discovery of user access patterns from Web servers, i.e., Web usage mining.

Web Mining is the extraction of interesting and potentially useful patterns and implicit information from artifacts or activity related to the World Wide Web.

There are roughly three knowledge discovery domains that pertain to web mining:
Web Content Mining,
Web Structure Mining, and
Web Usage Mining.

Web content mining is the process of extracting knowledge from the content of documents or their descriptions.

Web document text mining, resource discovery based on concepts indexing or agent based technology may also fall in this category.

Web structure mining is the process of inferring knowledge from the World Wide Web organization and links between references and referents in the Web.

Web usage mining, also known as Web Log Mining, is the process of extracting interesting patterns in web access logs.

There are two groups of web content mining strategies: Those that directly mine the content of documents and those that improve on the content search of other tools like search engines.

The subject of ontology is the study of the categories of things that exist or may exist in some domain.
An informal ontology may be specified by a catalog of types that are either undefined or defined only by statements in a natural language.
A formal ontology is specified by a collection of names for concept and relation types organized in a partial ordering by the type-subtype relation.
The applications of web mining are listed below

- Web Mining Application in E-commerce Customer Behaviour Analysis
- Web Mining Application in E-commerce Transaction Analysis
- Web Mining Application in E-commerce Website Design
- Web Mining Application in E-banking
- Web Mining Application in M-commerce
- Web Mining Application in Web Advertisement
- Web Mining Application in Search Engine
- Web Mining Application in Online Auction
- Web Mining Application in Online Knowledge Management
- Web Mining Application in Online Social Networking
- Web Mining Application in E-learning
- Web Mining Application in Blog Analysis
- Web Mining Application in Online Personalization and
- Recommendation Systems
- Web Mining and Intelligent Web Services
- Case and Empirical Studies

Model Question Paper
Paper 4.4 : Data Warehousing and Mining

Time : 3 Hours **Maximum Marks : 100**

PART – A **(5 x 8 = 40 marks)**

Answer any five questions
All questions carry equal marks

1. Explain the KDD process in Data mining
2. Write a brief note on OLAP technology
3. Discuss in detail about the Data mining primitives
4. Explain Information retrieval in detail
5. Discuss in detail about Decision Trees with suitable example.
6. Compare Classification and clustering
7. Discuss K-Means Clustering
8. Write a note on Web Content Mining

PART – B **(4 x 15 = 60 marks)**

Answer any four questions
All questions carry equal marks

9. Elaborate data mining functionalities, classifications and issues
10. Discuss in detail about association rule mining in databases
11. Describe the Dimensional modeling of Data
12. Explain in detail about the Chi-Square regression tests.
13. Discuss various clustering algorithms with suitable examples
14. What do you mean web mining? Explain about web usage mining and web mining applications
15. Discuss in detail about various partitional algorithms.

******

Educate ⇗ Empower ⇗ Elevate

Alagappa University formed in 1985 has emerged from the galaxy of institutions initially founded by the munificent and multifaceted personality, Dr. RM. Alagappa Chettiar in his home town at Karaikudi. Groomed to prominence as yet another academic constellation in Tamil Nadu, it is located in a sprawling and ideally suited expanse of about 420 acres in Karaikudi.

Alagappa University was established in 1985 under an Act of the State Legislature. The University is recognised under Sec. 2(f) and Sec.12(B) of the University Grants Commission. It is a member of the Association of Commonwealth Universities and the Association of Indian Universities. The University is accredited with 'A' Grade by NAAC.

The Directorate of Distance Education offers various innovative, job-oriented and socially relevant academic programmes in the field of Arts, Science, IT, Education and Management at the graduate and post-graduate levels. It has an excellent network of Study Centres throughout the country for providing effective service to the student community.

The distance education programmes are also offered in South-East Asian countries such as Singapore and Malaysia; in Middle-East countries, viz., Bahrain, Qatar, Dubai; and also at Nepal and Sri Lanka. The programmes are well received in India and abroad.



ALAGAPPA UNIVERSITY

(Reaccredited with 'A' Grade by NAAC)

KARAIKUDI-630 003, TAMILNADU

DIRECTORATE OF DISTANCE EDUCATION

(Recognized by Distance Education Council (DEC), New Delhi)

Sri Balaji Offset Press, Rjpm. Copies : 500

AU / DDE / D2 / Printing / Order 7 / 2015 Date: 17-02-2015